VAWKUM Transactions on Computer Sciences
http://vfast.org/journals/index.php/VTCS@ 2024, ISSN(e): 2308-8168, ISSN(p): 2411-6335
Volume 12, Number 2, July-December 2024 *pp:28-49*

Optimizing Machine Learning Classifiers for Credit Card Fraud Detection on Highly Imbalanced Datasets Using PCA and SMOTE Techniques

Zeeshan Ali Haider [1]*, **Fida Muhammad Khan** [1], **Abu Zafar** [2], **Nabila** [3], **Inam Ullah Khan** [1]

[1]Department of Computer Science, Qurtuba University of Science & Information Technology, Peshawar, Pakistan; [2]Department of Political Science, Government Fazlul Haque College, Chakar, Barishal, Bangladesh; [3]Department of Computer Science and Engineering, Southern University Bangladesh

Keywords: *Credit Card Fraud Detection, Machine Learning Models, SMOTE (Synthetic Minority Over-sampling Technique), Real-time Fraud Detection, Class Imbalance, Predictive Analytics.*

Journal Info:
Submitted:
October 02, 2024
Accepted:
October 12, 2024
Published:
October 20, 2024

Abstract

Card fraud detection refers to the process of identifying unauthorized or suspicious transactions made using credit or debit cards. It employs machine learning models, rule-based systems, and anomaly detection techniques to detect patterns indicating potential fraud. There is a growing need for systems that can accurately predict and prevent fraudulent transactions. Reducing financial loss by Implementing advanced detection models to safeguard it from fraud or malicious transactions. Therefore, we proposed machine learning models that will predict credit card fraud at an early stage. Also, the study used feature scaling, Principal Component Analysis (PCA), and the Synthetic Minority Over-sampling Technique (SMOTE) to deal with the class imbalance on the dataset. Moreover, SMOTE is applied to balance the classes by synthesizing examples of the minority class, making classifiers more robust. The results show that LR, SVM, KNN, and XGBoost models correctly predict 97% of fraudulent and non-fraudulent cases. The Decision Tree and the Random Forest models are capable of achieving at least 96%, respectively. This research combines advanced machine learning methodologies with real-time processing to give insights into predictive analytics in financial fraud detection, which may enhance accuracy and efficiency in financial security systems.

***Correspondence author email address:** Zeeshan.ali9049@gmail.com
DOI: 10.21015/vtcs.v12i2.1921

1 Introduction

Credit card fraud is a serious problem in worldwide financial development, particularly with the fast progress of online trading. In today's world for businesses from physical to digital payments, consumers are convenient but the transitions with the credit card have opened the door for new avenues of fraud. In 2018, the global organization losses of payment card fraud were estimated at nearly $27.85 billion by the Nilson Report which increases yearly [1]. The complete fact that it exists shows how as digital transactions have grown, the complexity and pervasion of fraud attempts have also increased presenting real implications for both consumers and financial entities [2]. Credit Card fraud detection is a difficult problem due to the absolute amount of monetary transactions processed daily, where manual and rule-based systems simply are not going to cut it. Legacy rule-based systems, based on established patterns and predefined thresholds, are proving outdated for fraudsters who continuously refine their methods[3]. Such systems are costly to develop and difficult to maintain with new fraud types and emerging trends and hence they can have high false-positive rates that hurt the consumers (due to wrong distrust in them) or for merchants who unnecessarily face higher operational costs [4]. As a result, research in more sophisticated, adaptive, and scalable fraud detection approaches has also received increasing attention.

The development of machine learning (ML) changed the landscape of credit card fraud detection providing a solution to the shortcomings of traditional methodologies [5]. Contrary to a rule-based system, machine learning models are trained on past data patterns and are then able to predict future transactions. This is especially useful in fraud detection, where new fraudulent schemes change all the time. Numerous studies over the past decade have shown various machine learning algorithms such as Decision Trees, SVMs and Neural Networks have performed with higher accuracy and lower false-positive rates in fraud detection [6]. One of the most groundbreaking disciplines is deep learning-based approaches to fraud detection. Deep learning is a branch of machine learning that uses neural networks with many layers to model complex, non-linear relationships in data. LSTM (Long Short-Term Memory) networks, a kind of Recurrent Neural Network (RNN), have been made use of to identify fraud in time-series data (e.g., consecutive transactions). These models can learn long-term dependencies and catch small anomalies that indicate fraudulent activity [7]. In addition, the advancement of fraud detection systems is heightened through the integration of big data technologies into machine learning models. This makes the volume of data generated by financial transactions enormous and provides a goldmine of information to identify fraud when accurately analyzed [8]. Big data frameworks enable us to process an enormous volume of unstructured, and semi-structured data so that fraudulent activities can be identified immediately. This is especially critical in the case of credit card fraud where delay in detection may result in huge monetary loss [9].

The merger of big data and machine learning makes it possible for better features can be engineered to train fraud detection methods [10]. For instance, transaction history and customer profile features or other external data sources (e. g. social media activity) make the model specifics stronger for the fraud / non-fraud transaction classification model [11]. This mode of operation results in better detection accuracy and fewer false positive alerts, allowing legitimate transactions to be processed with minimal interruption. As electronic payment volume grows, especially credit and debit card use, fraud has become a major issue undermining financial security worldwide [12]. One of the keys to making sure consumers and financial institutions stay protected from fraud, something that can result in significant financial losses and loss of face respectively [13], is efficient detection with real-time processing. This has given rise to significant research and implementation of systems that can provide credit card fraud detection mechanisms employing CAD or Computer Aided Detection methods [14].

However, with the rise of digital transactions, especially online, scammers and fraudsters now have an even bigger playground, causing credit card fraud to increase too [15]. By the late 2010s credit card fraud numbers were running into the billions This was a considerable sum of money for networks and other organizations to find themselves at financial risk [16]. These statistics have given rise to dramatic advances in the techniques deployed

for fraud detection and prevention One such change is the increasing use of machine learning which has led to a shift in how fraud detection tools can be dynamic, adaptive, and more effective [17]. The earlier conventional methods, such as rule-based systems used hardcoded logic and fraud patterns to detect fraud which led to a high rate of false positives and these could not keep up with the evolving nature of fraud [18]. However, the development of machine learning and artificial intelligence in time pushed researchers and practitioners to develop more sophisticated models that can learn and adapt over time to changing patterns in data [19]. Publications have been made demonstrating the ability of ML methods such as DTs, SVMs, and NNs to enhance detection rates with a particular aptitude to reduce the rate relating to false positives [20].

Deep learning has provided new weapons in this constant battle against credit card fraud. Specifically, neural networks particularly those that leverage LSTM cells have been well-positioned to identify subtle and complex patterns from transaction data that may elude other models [21]. These methods can handle high-volume analyses of data, they can learn from subsequent transactions and keep up with fraud pattern changes significantly well [22]. With more and more financial transactions being conducted online it is vital to have antifraud systems that can handle a massive amount of traffic that gets continually attacked. Payment providers and financial institutions need such risk analysis systems to retain the security of digital commerce as well as consumer trust in electronic payment systems [23]. The future trends in fraud detection are expected to exploit the following levels of improvements, such as improvement in machine learning and artificial intelligence and innovation for data collection and processing big data technologies and cloud computing [24]. Continued research in this field is fundamental in the journey of advancing credit card fraud detection, an essential element of contemporary financial security models that bridge technological innovation with years-long traditions, concerning digital age trust and safety.

With digital financial transactions increasing in popularity, the issue of fraud on the internet as well as over the phone grows at a quick pace and criminals find ways to be more sophisticated about going after their targets. This system as part of the project ameliorates the pressing requirement for sophisticated, machine learning-based fraud alarm management systems that are capable of considerably minimizing fraudulent acts. The achievement is to make a model that can learn transactional patterns and behaviors to accurately discover fraud while also keeping the false positives and negatives as less likely outcomes. A simulated environment will be used to evaluate and improve the model to ensure it is dynamic enough to respond to new breeds of fraud. In the long run, improving transaction security and loss minimization will increase trust in digital platforms by end-users. The key contributions of this research study are:

- To formulate a model that learns from transaction data to keep up with emerging fraud patterns and increase the accuracy of detection.
- To execute a fraud detection model using various ML algorithms with hyper-parameter tuning and pick the best classifier based on accuracy.
- To test the efficiency of the model in discovering different kinds of frauds in a diverse range of transaction environments, maintaining high precision and low false negative rates. To evaluate how quickly the model fits into those existing systems to detect and respond to fraud in real-time without inhibiting transaction flow for legitimate transactions.

The rest of the paper is structured as follows: Section 2 provides literature reviews and a detailed look at methods used in fraud detection, and Section 3 outlines steps for data preprocessing, feature engineering, and application of different machine learning algorithms with model evaluation and tuning leg thereof. Section 4 presents the experimental results & discussion of the outcomes of applying these techniques to our dataset will be shown, including model performance comparisons, and then the paper ends with conclusions, directions for

financial security, and future research in Section 5.

2 LITERATURE REVIEW

This study of literature review goes into some depth on the existing work demonstrating the effectiveness of machine learning techniques in credit card fraud detection. This encompasses a wide variety of methods from traditional algorithms such as DTs and SVMs to ensemble methods that effectively reduce false positive rates in imbalanced datasets, like Random Forests or Gradient Boosting Machines. The review also explores more advanced approaches, such as deep learning-based LSTM networks which can detect complex patterns and anomalies in transaction data. It also discusses unsupervised learning for fraud detection in low-label data scenarios and the influence of big data technologies and AI advances in real-time fraud detection [25]. The review identifies current failures and limitations in the systems, paving the way for the utilization of machine learning to more accurately make sense of these fraud detection systems and how machine learning can augment them for greater precision.

The rapid digitalization of financial transactions has also led to a huge growth in credit card frauds which in turn have become a serious threat for both the banks as well as customers. In the past, fraudulent activities were mainly detected with rule-based systems; they worked based on pre-set rules to identify suspicious transactions. These policies were often rules based on simplistic heuristics such as transaction amounts greater than a certain dollar value or transactions in unexpected geographies[26]. Although these systems offered an initial line of defense, they were static and could not accommodate the constantly changing threat landscape that fraudsters used. Such systems were extremely inflexible, and their rigidity often resulted in false positives: legitimate transactions that were identified as fraudulent and thus interrupted normal operations both for customers (in terms of user experience) and financial institutions (extra costs) [27]. Also, with improved fraud tactics happening, the rule-based systems became harder to manage as there were more unseen fraudulent activities.

However, the trend in recent years which more and more people are following is to use computational powers like ML and DL algorithms to improve fraud detection. It has a massive advantage in fraud detection because it can learn and detect complex patterns from data, thus its application (machine learning) has changed the game of fraud detection. With the help of ML models, unlike rule-based systems, these are designed to detect fraud by identifying known and unknown behavior patterns that have historically been used by bad actors [28]. Unlike static rules that rely on signal detection, these models can learn over time and adjust as new data comes in which better protects against the introduction of fraud techniques. For example, techniques like ensemble-based methods such as Random Forests and Gradient Boosting Machines have become very popular for fraud detection due to their good performance with imbalanced datasets [29]. This ability is critical in fraud detection as the number of fraudulent transactions is minuscule compared to the legitimate ones. In other words, you can create ensemble methods to enhance the accuracy and robustness of your binary classifier by the predictions of multiple base models which contribute to fewer false positives and negatives.

In the realm of fraud detection, deep learning (a subfield of machine learning) has taken things to a whole new level as well, capable of modeling intricate and non-linear relationships in data. Neural networks in particular, those using architectures like Long Short-Term Memory (LSTM) networks have delivered outstanding results for numerous problems such as language modeling and more recently word embeddings. As an example, LSTMs perform well on sequential data like transaction histories, the order of transactions can give significant context to flag something as anomalous [30]. For instance, LSTMs can make use of temporal dependencies to spot patterns that may alert suspicious activity (e.g., a burst of transaction frequency or an irregular sequence of purchases). In addition, DL models are capable of handling large amounts of data in parallel making them ideal for computer vision applications. The capability to process the data as it is generated makes sure that incoming fraudulent transactions can then be pinpointed and denied, thus minimizing the possibility of financial failure.

Additionally, unsupervised learning approaches in fraud detection are said to be a key component due to the scarcity of labeled data as well pointed out unsupervised learning techniques, including clustering and anomaly detection, do not require a pre-defined set of labels with which to perform fraud identification. Instead, these techniques try to find the normal structure of data and point out transactions having huge deviations from normal distribution [31]. Clustering techniques, for instance, can cluster transactions based on their similarity to one another so that we can see outliers that may represent fraud. These ensemble classifiers can either also be used as semi-supervised (detecting outliers) or for managing transactions that seem out of the pattern detected by another algorithm, they are suspicious, and further investigation is necessary. Although unsupervised learning methods are not as accurate as supervised methods, they are important in the first steps of fraud detection where its purpose is to reduce a great breach of potentially fraudulent transactions for further analysis [32]. Apart from the progress in machine learning and deep learning, big data technologies have led to more efficient fraud detection systems. With digital transactions rising exponentially, the necessity to process and analyze large amounts of transaction data in real-time has taken a critical role. Big data technologies are capable of processing large loads of transaction streams in real-time so that suspicious and fraudulent activities can be identified and prevented immediately [33]. This ability is especially important when related to online and mobile payments, as the quicker fraud can be detected, the better experience a user has while also further protecting them financially. Fraud detection in real-time will enable the system to automatically identify suspicious transactions and block them, thus limiting fraud activities and avoiding potential losses.

Furthermore, the machine learning models worked into big data platforms have offered a way for fraud detection systems that are more sizeable and boast a better ability to detect in real-time [34]. Modern fraud detection systems made some progress in that way, for instance, leveraging distributed computing frameworks such as Apache Spark and Hadoop to deal with the huge data processing demand on them. A real-time Process Framework allows to processing of transaction data in parallel across multiple nodes which can accelerate machine learning model training and model deployment time-saving [35]. Additionally, being able to deploy fraud detection systems seamlessly at scale on cloud computing platforms has given financial institutions more elasticity for scaling and descaling their operations as required. The literature also reveals the increasing practice of integrating artificial intelligence (AI) technologies in fraud detection systems. AI, particularly reinforcement learning and neural architecture search-based methodologies could facilitate building even more adaptive and precise fraud detection models [36]. For example, reinforcement learning allows a model to learn the optimal strategy of fraud detection through trial and error, constantly improving its performance. In contrast, besides handcrafted designing of the neural network architecture for a defined fraud detection task, neural architecture search techniques can automatically construct optimal model architectures and that too application-specific highly tuned models.

While this is all with a steady rise in technology, the fraud and misuse detection process remains a challenge due to continued innovations and changes in fraudulent activities. Evolution of Fraud Detection: As fraudsters continue to adapt and shift their tactics, so must the systems be detecting those very attempts. This means that an increasing focus is given to adaptive fraud detection systems which can react in real-time to adapt to new threats [37]. More and more research in this field is aimed at increasing the interpretability and explainability of machine learning models, thereby allowing the financial industry to trust the decisions made by these systems and understand their justification. Over the years, the literature on credit card fraud detection has evolved along with trends in artificial intelligence, data science, and big data. This transition from old manual rule-based systems to these new, highly sophisticated ML and DL methodologies provides stronger fraud detection models that are designed to cater the modern financial transaction complexities. This has been improved to the next level by integrating big data with real-time processing so we can detect if a fraud activity can happen in 100s of milliseconds. Nonetheless, the evolution of fraud is ongoing and thus further research and innovation in this key sector

are still crucial. The future of fraud detection research likely lies in the creation of even more adaptive, scalable, and interpretable models and in taking advantage of oncoming technologies like blockchain, and AI to improve security and prevent fraud.

In our study, we proposed multiple machine learning algorithms to improve their precision in this type of data and extend upon these results. It calculates values for some of the metrics such as Model Parameters to improve and uses advanced techniques to increase accuracy in fraud detection in credit card transactions. We want to build a new system that affords more precision in identifying acts of fraud, and subsequently fewer false alarms than the status quo offers. Our target is now parity with contemporary fraud detection methods. The literature review is a broad examination of credit card fraud detection methods with the gradual shift from traditional techniques to machine learning models. This paper talks about decision trees, support vector machines, and ensemble methods to detect the given dataset in detailing with effectiveness of algorithms in dealing with imbalanced datasets as well as its improvement in detection coverage. It also highlights the importance of deep learning, especially Long Short-Term Memory (LSTM) networks which can capture long sequences of transaction data and model such complex patterns and anomalous behavior. It also elaborates on the application of unsupervised learning for fraud identification when there is very little labeled data available, and the role of big data technologies and recent AI advancements for real-time processing. Current methods were reviewed, noting gaps and challenges that provide precedence for our study to improve model performance range through tuning and optimization for enhanced fraud detection in credit card transactions.

3 Material and Methods

In this section, we defined the dataset, preprocessing steps, and ML techniques applied in building a fraud detection model. This research study described how to classify fraud or non-fraud transactions in credit card records, which are unbalanced data.

3.1 Proposed Architecture

The proposed architecture of credit card fraud detection uses a structured pipeline, concatenating preprocessing, model training process, evaluation, and real-time deployment. As a first step, the raw transaction data is preprocessed and used for feature scaling and dimensionality reduction using Principal Component Analysis (PCA) for better-quality data. In this research paper, to overcome the imbalance problem, synthetic samples for the minority class are generated by making use of the Synthetic Minority Over-sampling Technique (SMOTE), to obtain balanced training data. After processing the data, it is divided into training and testing sets, where different machine learning algorithms such as Logistic Regression, SVM, Decision Tree Random Forest K-Nearest Neighbors (KNN) XGBoost are trained and assessed. All models are further fine-tuned using hyperparameter optimization to optimize the performance metrics including accuracy, precision, recall, f1-score, etc. This architecture enables a fraud detection system powerful enough to address new fraud patterns, and scalable to adapt. Figure 1 illustrates the proposed architecture design for credit card fraud detection.

3.2 Credit Card Fraud Dataset

This research study is based on the "Credit Card Fraud Detection" dataset, one of the well-known resources for fraud detection. The dataset contains transaction details from European cardholders, with identity information being deliberately complicated. The data in the dataset are sensitive due to privacy concerns therefore the data in the dataset are given in such a format that is not understandable because the features are encrypted in the dataset to avoid unusual activity. The dataset is unbalanced and does not have a significant number of fraudulent transactions relative to other records. This imbalance causes a problem during model training and evaluation, which requires paying close attention to how it is handled and processed. Table 1 shows and explains the dataset parameter along with a description.

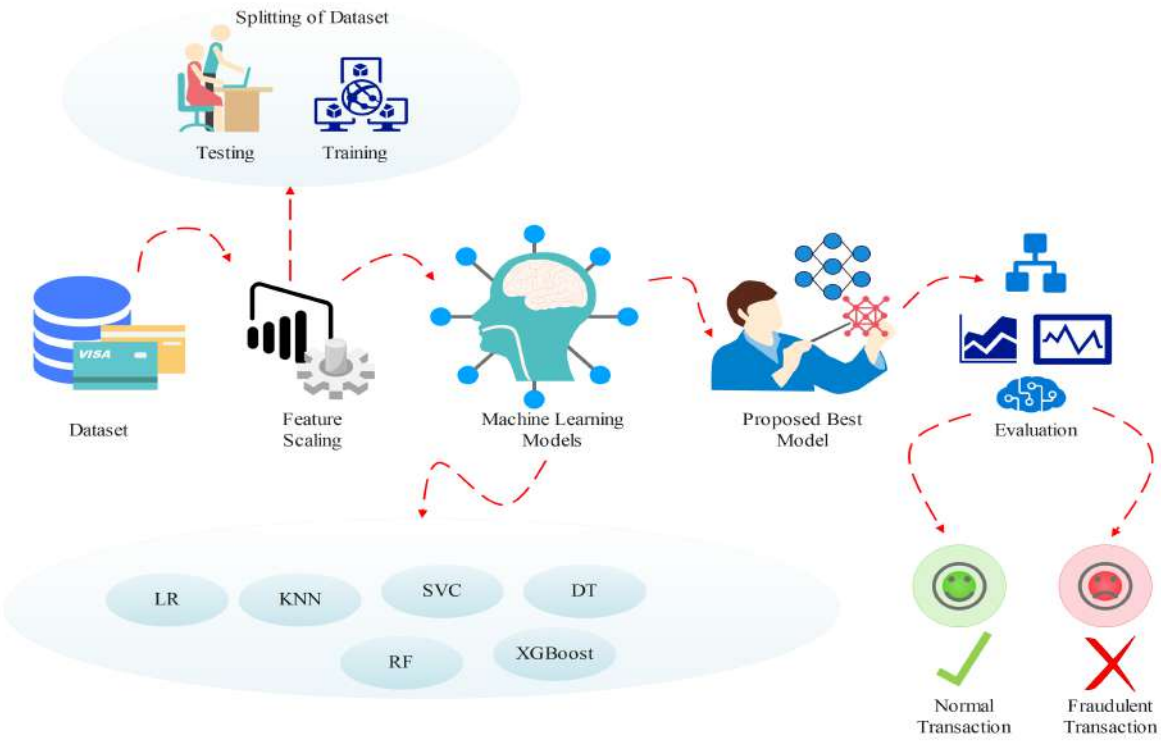

Figure 1. Proposed Architecture

PCA is used for transforming the data into a lower-dimensional form for anonymized features (V1, V2, ... V28) while preserving the most important patterns of data. The Amount represents how much money is in the transaction and the Class represents the outcome whether the transaction is fraudulent (1) or not (0).

3.3 SMOTE Technique

We apply the Synthetic Minority Over-sampling Technique (SMOTE) to tackle with class imbalance problem in the credit card fraud dataset. This is very important because the dataset is highly unbalanced i.e. there will be relatively few fraudulent transactions among a vast majority of normal ones. SMOTE is a technique used to produce synthetic samples for the minority class (fraudulent transactions) to balance the dataset and hence make the data more balanced so that machine learning models can be trained with maximum efficiency.

3.4 Dimensionality Reduction

To overcome the high-dimensional issues of the dataset, the Principal Component Analysis (PCA) technique is implemented. PCA is a statistical process to convert the original data set features into new variables referred to as principal components. The components are orthogonal, i.e. they are linearly uncorrelated. First of all, PCA is aimed at reducing the dimensions of the dataset but managing to retain as much variance as possible in fewer features. PCA reduces the spice dataset by projecting it onto a few principal components that explain most of the variance. Dimensionality reduction lessens the risk of overfitting and also mitigates computational complexity thus shortening the training time that is needed to learn a more accurate model.

3.5 Feature Scaling

Feature scaling is one of the most important steps to get perfect results in the step of training a machine learning model. Through this process, we make sure that each feature has an equal weight in the model which is

Table 1. Dataset Parameters

Parameter	Description	Parameter	Description
Time	The number of seconds between this transaction and the first transaction in the dataset	V16	Sixteenth anonymized PCA feature
V1	First anonymized PCA (Principal Component Analysis) feature	V17	Seventeenth anonymized PCA feature
V2	Second anonymized PCA feature	V18	Eighteenth anonymized PCA feature
V3	Third anonymized PCA feature	V19	Nineteenth anonymized PCA feature
V4	Fourth anonymized PCA feature	V20	Twentieth anonymized PCA feature
V5	Fifth anonymized PCA feature	V21	Twenty-first anonymized PCA feature
V6	Sixth anonymized PCA feature	V22	Twenty-second anonymized PCA feature
V7	Seventh anonymized PCA feature	V23	Twenty-third anonymized PCA feature
V8	Eighth anonymized PCA feature	V24	Twenty-fourth anonymized PCA feature
V9	Ninth anonymized PCA feature	V25	Twenty-fifth anonymized PCA feature
V10	Tenth anonymized PCA feature	V26	Twenty-sixth anonymized PCA feature
V11	Eleventh anonymized PCA feature	V27	Twenty-seventh anonymized PCA feature
V12	Twelfth anonymized PCA feature	V28	Twenty-eighth anonymized PCA feature
V13	Thirteenth anonymized PCA feature	Amount	Transaction monetary units
V14	Fourteenth anonymized PCA feature	Class	Whether the transaction was fraudulent (1) or not (0) in the target variable
V15	Fifteenth anonymized PCA feature		

essential when training algorithms that are very sensitive to the scale of data. Feature scaling techniques like Min-Max Scaling or Standardization (Z-score normalization) are used to standardize the range/distribution of every feature. Min-max scaling, which scales the feature to a fixed range usually 0s and 1s, and Standardization, which transforms the features by setting their mean as 0 and standard deviation as 1. Here is an algorithm that needs to be scaled, K-Nearest Neighbors (KNN), and Support Vector Classifiers (SVC) one of the reasons is that they are distance-based algorithms, so each feature has its scale you could eventually get a mask or obscure patterns out the result. Standardizing the features makes these algorithms more robust and gives us accurate model development with meaningful results.

3.6 Splitting of Dataset

To systematically check the performance of this machine learning model into unseen data, we follow a procedure, scrape data on Yelp reviews, and specifically divide it into a training and testing set. Usually, about 70% of the data is used to train, and then the model is tested on 30%. A training set is used to train the model, enabling it to find patterns and purposes in the data. A testing set that has not been seen just to evaluate the model in unseen data. This separation is important and used to Test the generalizability of our model beyond the training data set, in other words testing its performance measures exhibit or generalize how it will perform on real-world data.

3.7 Machine Learning Models for Model Training

The model is trained and evaluated using multiple machine-learning algorithms to classify fraudulent and non-fraudulent transactions. In this research study, we used the following machine learning models for credit card fraud detection and prediction.

- **Logistic Regression (LR):** Logistic Regression is a statistical model that is used to predict only two classes of labels. It calculates the probability of a binary outcome by applying a logistic function to the linear combination of input features. The method is simple and interpretable so you can use it as a good reference point for the models throughout the article.
- **K-Nearest Neighbors (KNN):** K-Nearest Neighbors is a non-parametric classifier that predicts the class of a given test observation by identifying the majority class of its nearest K data points in feature space. Its core idea lies in the calculation of distance between a test instance and its K-nearest training examples followed by assignment of most common class label among these neighbors to that test instance. KNN is easy to implement and effective for a lot of problems, but it can be whinny when it comes to large datasets.
- **Support Vector Classifier (SVC):** The Support Vector Classifier tries to identify the best hyperplane that passes via a set of classes in function area with most margin SVC allows us to use kernel functions that transform the feature space into a higher dimension in order to model non-linear relationships. The advantage of SVM is that it works very well in high dimensional spaces and when we have clear margin of separation between the two classes.
- **Decision Tree (DT):** With Decision Tree all the data is divided into subsets based on the input feature values which generates a tree structure of decisions. A tree has nodes for decision rules and its edges are outcomes of those decision rules. Decision Trees have interpretability but they tend to overfit, especially for complex trees.
- **Random Forests (RF):** It is an ensemble learning method based on the bagging and its combines multiple decision trees in order construct a stronger model. Aggregate over the output of multiple decision trees, enables the Random Forest to effectively reduce the risk of overfitting and thus generalizes better.
- **XGBoost:** XGBoost (eXtreme Gradient Boosting) is an advanced technique using gradient boosting algorithm, that builds models sequentially in the form of trees. Every new tree corrects mistakes that were made by the previous trees with an aim to improve predictive performance. XGBoost is deemed one of the most powerful gradients boosting trees that offer top-of-the-line performance in accurate predictions, scalability, and fast runtime.

3.8 Proposed Best Model

The model with the best performance based on the initial evaluation of various machine learning algorithms is further chosen in terms of comparison of performance metrics. AUC-ROC is one of the most important criteria for model selection and it involves Accuracy, Precision, and Recall. The most suitable model for fraud detection is the one that attains the best compromise between these metrics and it provides treatment to class imbalanced data of

credit card fraud dataset. We Leading Artificial Intelligence Development Company offer hyperparameter tuning and model optimization which is a significant activity to make your AI model perform well. This means tuning parameters to maximize the fraud detection rates with minimal false negatives. A model with high performance on various metrics as well as robustness against the inherent imbalance in the dataset, to alleviate the problem of overfitting and improve fraud detection overall

3.9 Model Testing

The final stage of the research is to test your chosen model using the reserved test set to see how well its performance holds up in real-world use cases. This is the phase when a model can detect whether a transaction is fraudulent, not looking at the precision from the point of time. Evaluation: In evaluation, the model reflects metrics like Accuracy, Precision, Recall, F1-score, and AUC-ROC. Precision measures the percentage of actual frauds out of all transactions that we have flagged as fraudulent. Recall is a measure of how well the model can detect every single real fraudulent transaction. F1-score, therefore, being a harmonic mean of precision and recall gives us a single portion to assess the model performance. AUC-ROC assesses the model for its ability to differentiate classes at various thresholds. The models are analyzed as the percentage of fraud it is capable of detecting (true positives) and also the fraudulent transactions we missed when they occurred (false negatives). The goal should be to have the model pick out as many fraudulent transactions as possible so they are caught on time but also avoid identifying false positives (good or normal transactions) thus making the fraud detection system more reliable.

4 RESULT AND DISCUSSION

The experimental results of credit card fraud detection are described in this section. Experimental results were carried out using Google Colab resources to investigate different machine-learning approaches, namely LR, KNN, SVC, DT, RF and XGBoost . We also leverage the principal component analysis (PCA) technique for dimensionality reduction and SMOTE for handling class imbalance. These models have generated output results which have been summarized in detail with regards to the performance metrics of these models and comparison analysis on who each model is effective in detecting fraudulent transactions.

4.1 Loading the Dataset and Necessary Libraries

We load different necessary libraries for data preprocessing, visualization, and manipulation. We used Pandas library which highlights and analyzes data structure, and NumPy which is used for numerical operations and array handling. We used Matplotlib for visualizations and Seaborn which makes the visualizing of data built on top of Matplotlib. We used Pandas Library to load the dataset of Credit Card Fraud detection. Scikit-learn is used for the implementation of machine learning algorithms, as well as data shuffling and model evaluation. Data stored is randomized using the random function and import shuffle from sci-kit-learn. A gradient boosting framework, XGB, is further installed to carry out efficient model training, and utility functions are imported for different tasks. It uses the train test split function from Scikit-learn to split the dataset into training and testing sets, whereas StandardScaler scales feature values. We evaluate the model by accuracy, classification reports, confusion matrix, precision-recall curve F1 score, and AUC.

4.2 Visualization of the Dataset

Using Seaborn and Matplotlib popular Python libraries for data visualization we plot a histogram of the 'Time' variable from the dataset. We plotted the distribution of transaction times, with an overlayed KDE (Kernel Density Estimate) for a smooth graph distribution of Transaction Time, shown in Figure 2. Finally, through the show () function, we display a histogram that shows how many times different durations between transactions occur in the dataset.

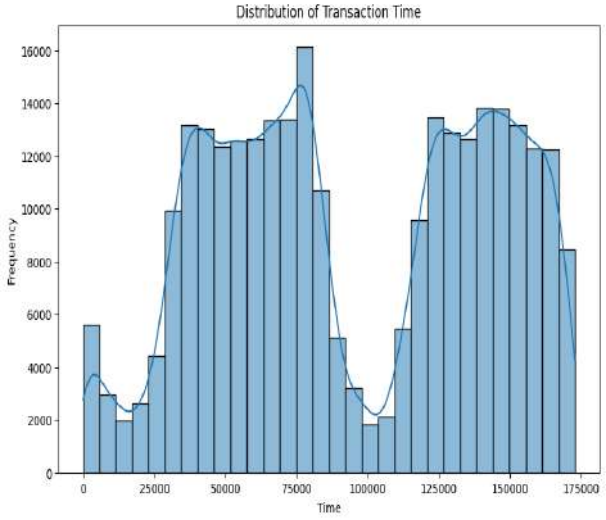

Figure 2. Distribution of Transaction Time

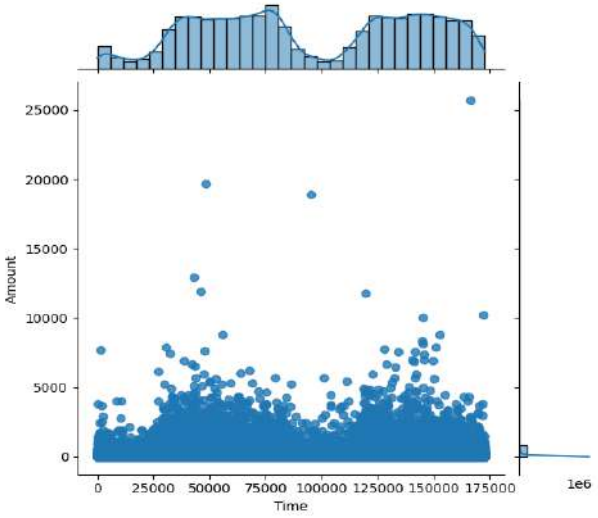

Figure 3. Relationship between Time and Amount

We also visualize how 'Time' and 'Amount' from the data set relate to each other using Seaborn and MatPlotLib. Figure 3 shows that the transaction amount is distributed over time. We customize these by setting the number of bins to 30 and also filling them in for a better visualization using marginal kws. The plot is then displayed using show () with a more in-depth look at the relationship between the transaction time and amount.

We visualize the correlation between different features in the dataset using a heatmap. A heatmap is then generated to represent the correlation matrix of the dataset as shown in Figure 4, where each cell in the heatmap corresponds to the correlation coefficient between two variables. The plot is shown using plt.show(), given a pure visual illustration of how features in the dataset relate to one another.

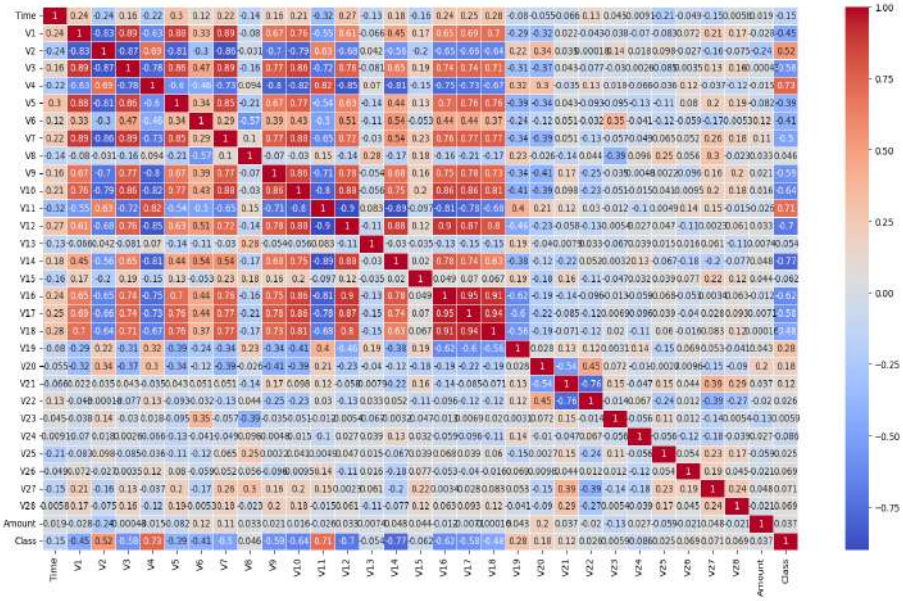

Figure 4. Correlation Matrix

4.3 Data Preprocessing and Dimensionality Reduction for CCF Detection

We divided our dataset into training and testing sets using the function train test split, test size was set to 0.3 to allocate 30% of data in testing, and random state was used as 42 for reproducing ability intensity. Then use the standard scaler to scale certain columns "Time" and "Amount". The features are made more comparable by scaling, which is generally a good thing as the data after standardization may be in order and is great for further analysis. Since earlier we transformed two columns, after scaling these columns are merged with the other features in the dataset and this creates a new DataFrame (d scaled). The data dimensionality is then reduced using PCA, we set the number of components to 7 and create a PCA object, transforming the scaled data into these lower dimensions. An important point to note is that we have two arrays available now, explaining variance ratios and values of the principal components, which helps us get an idea about the portion of variance contributed by each component. Change it to a DataFrame with new feature names and combine the target variable 'Class' to create new data. Last but not least, we print the shape of new data DataFrame and save it in CSV as final data. CSV for the data that will be used in modeling/prediction with dimensionality reduction.

4.4 Hyperparameter Tuning of Logistic Regression Model

We import the libraries required for the Logistic Regression model. It is used as a reading object of the LR model which fits the training data (X train, y train). After training, we estimate the test data (X test) with the trained model. The code then uses the confusion matrix function to determine various metrics about the model performance true positives, false negatives, true negatives, and false positives. Confusion Matrix helps to get with precision and types of error in the model. The code uses GridSearchCV to find the best parameters for our Logistic Regression model with hyperparameter tuning. GridSearchCV uses the combination of over given parameters, parameter search space penalty types (L1 and L2), and its regularization strengths (c values from 0.001 to 1000) to create parallel sets of parameters which will all be applied on a linear SVM model for the classification task. These are the best parameters found by the GridSearch, and then we predict them with our test data. Finally, print the classification report for the tuned model predictions. The report has metrics including precision, recall, F1-score, and support for each class as shown in Table 2.

Table 2. Classification Report of Logistic Regression

Class	Precision	Recall	F1-Score	Support
0	0.96	0.99	0.97	626
1	0.98	0.95	0.97	574
Accuracy			0.97	1200
Macro Avg	0.97	0.97	0.97	1200
Weighted Avg	0.97	0.97	0.97	1200

4.5 Confusion Matrix of Logistic Regression Model

We determine a sequence of steps that we follow to train and evaluate a Logistic Regression model in the context of credit card fraud detection. It starts by importing pandas for data manipulation and numpy for numerical operations, alongside the visualization libraries Seaborn and Matplotlib. Data visualization using pyplot also imports machine learning tools from sklearn, like train test split for data splitting, Standard Scalar for feature scaling, PCA for dimensionality reduction, and LR for modeling. Setting DataFrame with the dataset, its values are taken from a CSV file and shuffled to make random. It then reshapes the two classes of data (data that is fraud, or data that is non-fraud), here it constrains the sample size of data without any fraud to 2000, and mixes them up (card faces down) using a Weighted Random Sampler. To deal with the imbalanced data, oversampling is carried out through

the SMOTE technique which generates synthetic samples for the minority classes. We apply feature scaling to the 'Time' and 'Amount', and then Principal Component Analysis (PCA) is applied to the data. The data set is divided into training and testing sets with a ratio of 70:30. The LR model is fitted into the training data and evaluated on the test set. At the end, a confusion matrix is created to check the performance of the model and visualize it using a heatmap. The visualization of true positive, true negative, false positive, and false negative is shown in the matrix hence we can have a clearer understanding of how well our model was able to detect fraudulent transactions and where it is making mistakes as shown in Figure 5.

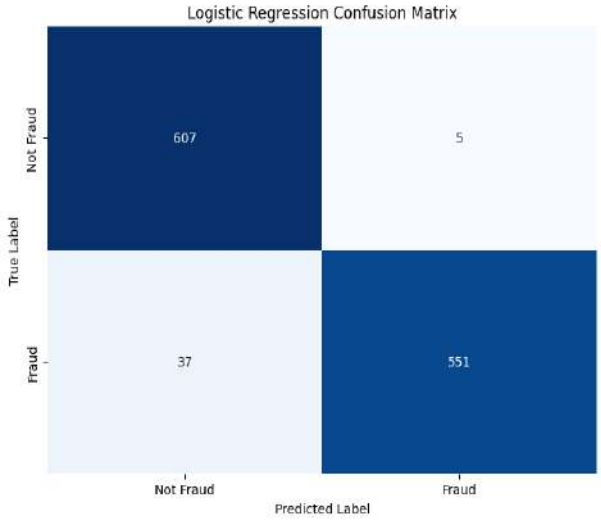

Figure 5. Confusion Matrix of Logistic Regression

4.6 Hyperparameter Tuning of Support Vector Machine Model

We Initiate an SVM model with RBF kernel and train on the training set. The model is then used to make predictions on the test set and output performance metrics such as the classification report and confusion matrix to evaluate it.

To increase the performance of the model, we used Grid Search with Cross Validation (GridSearchCV) which implies finding the most suitable hyperparameters such as regularization parameter, type of kernel, and gamma. Grid search determines the best parameters concerning the accuracy and gives us the suitable parameter values for the SVM model. The code then creates an SVM model with these parameters (C=100, gamma=0.01, kernel='rbf') and re-fits it over the data again. Finally, the optimized model is used to make predictions on the testing set again and then print an updated classification report as shown in Table 3, so that the model is evaluated against the tuned hyperparameters giving us better accuracy and confidence on the fraudulent transaction detection.

Table 3. Classification Report of SVM

Class	Precision	Recall	F1-Score	Support
0	0.95	0.99	0.97	612
1	0.99	0.95	0.97	588
Accuracy			0.97	1200
Macro Avg	0.97	0.97	0.97	1200
Weighted Avg	0.97	0.97	0.97	1200

4.7 Confusion Matrix of Support Vector Machine Model

To check the model's performance, we compute the confusion matrix to see how well it classifies the fraudulent and non-fraudulent credit card data. We plot the heatmap and annotate it with counts for TF, TN, FP, and FN while displaying it with a support color map (Blues) to make it more readable as shown in Figure 6. This visualization helps to study the performance of the SVM model in detecting whether a transaction was fraudulent or not.

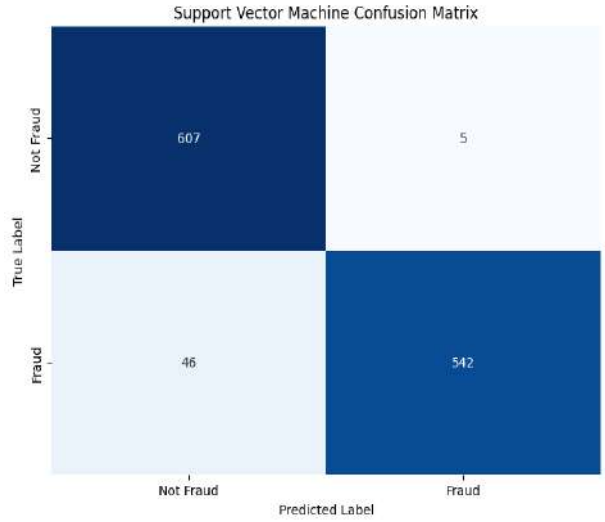

Figure 6. Confusion Matrix of SVM

4.8 Hyperparameter Tuning Decision Tree Model

Once this model is trained it is used to predict the test set and the DT model to evaluated with a classification report and confusion matrix. It gives detailed metrics like Precision, Recall, F1-score also TF, TN, FP, and FN counts. We perform hyperparameter tuning to improve the model accuracy and then carried out with GridSearchCV. Grid search will try all different combinations of parameters such as criterion to split ('Gini' or 'entropy'), max depth of the tree, minimum samples required to be at a leaf node, etc. The grid search fits a separate model on the training data and uses this new Decision Tree model to evaluate the test set. The performance of the updated model is summarized with a classification report to understand how better hyperparameter tuning has been performed as shown in Table 4.

Table 4. Classification Report of Decision Tree

Class	Precision	Recall	F1-Score	Support
0	0.95	0.97	0.96	612
1	0.97	0.95	0.96	588
Accuracy			0.96	1200
Macro Avg	0.96	0.96	0.96	1200
Weighted Avg	0.96	0.96	0.96	1200

4.9 Confusion Matrix of Decision Tree Model

A Confusion matrix is plotted as a heatmap to understand the performance of the decision tree model. Using Seaborn and Matplotlib to create the heatmap allows for a visual representation of how accurate the model is and

where its errors are, understandably, providing us with an idea of which part it failed in distinguishing fraudulent from non-fraudulent transactions as shown in Figure 7.

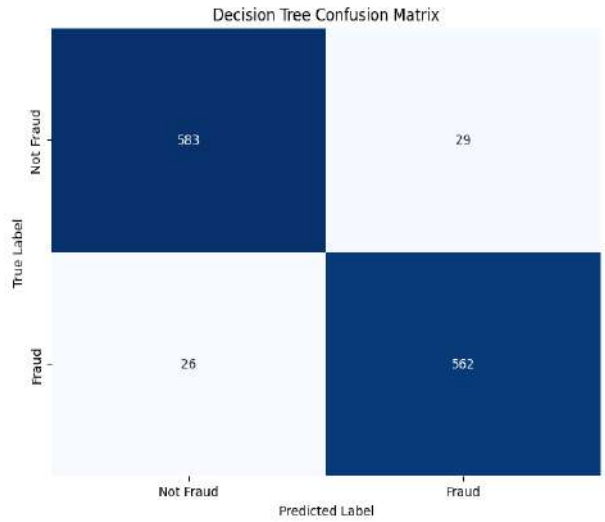

Figure 7. Confusion Matrix of Decision Tree

4.10 Hyperparameter Tuning of Random Forest Model

Random Forest Classifier from sklearn.ensemble Credit card fraud detection using ensemble building and evaluation of random forest model. Here, Random Forest Classifier is constrained to n estimators 5 which means that the ensemble will use 5 decision trees. The model is then fit on training data X train, y train, and predictions are generated from test data X test. We evaluate how well the random forest model works by using a confusion matrix and classification report. The TP, TN, FP, and FN counts are given by the confusion matrix, whereas the classification report provides detailed precision, recall, F1-score, and accuracy of each class as shown in Table 5. These metrics allow us to evaluate how well the random forest model can separate fraudulent and non-fraudulent transactions.

Table 5. Classification Report of Random Forest

Class	Precision	Recall	F1-Score	Support
0	0.95	0.98	0.96	612
1	0.97	0.94	0.96	588
Accuracy			0.96	1200
Macro Avg	0.96	0.96	0.96	1200
Weighted Avg	0.96	0.96	0.96	1200

4.11 Confusion Matrix of Random Forest Model

A Random Forest Classifier from sklearn. use the ensemble to create a random forest model for fraud detection. We configure the model to have n estimators 5, which means this ensemble contains 5 decision trees Next, this trained model is used to predict the testing data (X test) from the historical data by passing X train and y train as training data. The confusion matrix shows the model performance assessment and the matrix is then displayed in a heat map using both Seaborn and Matplotlib, establishing the counts of TP, TN, FP, and FN as shown in Figure

8. This color gradient heatmap shows how the random forest model can separate fraudulent transactions from non-fraudulent transactions.

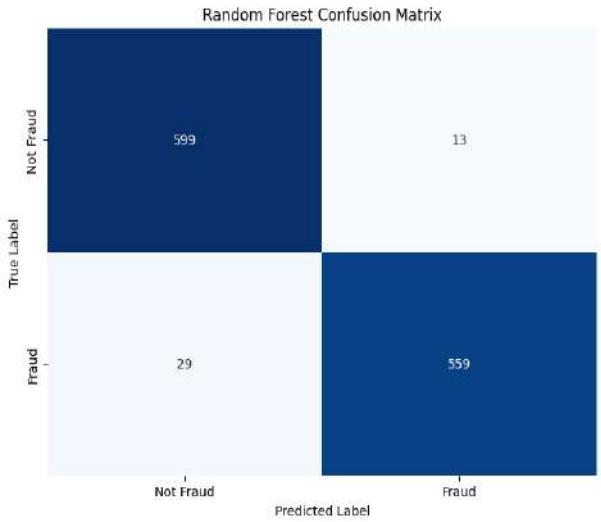

Figure 8. Confusion Matrix of Random Forest

4.12 Hyperparameter of Tuning KNN Model

The first argument n neighbors is an extracted value from the KNN model which means the number of nearest neighbors that will be taken into consideration, when we are going to classify a data point. The model is trained on the training set (X train and y train), and tested using the test set (X test). The confusion matrix and classification report as in Table 6 shows the performance of the model, which describes the accuracy, precision, recall, and F1-score of the model. Hyperparameter tuning is performed to optimize the KNN model using GridSearchCV. The grid search results are then used to determine the best parameters. We train a final KNN model with n neighbors 2, and evaluate it on the test set. Results, confusion matrix, and classification report after multiple K values and algorithm are indicated to compare performance across different values of K.

Table 6. Classification Report of KNN

Class	Precision	Recall	F1-Score	Support
0	0.95	0.99	0.97	612
1	0.99	0.94	0.97	588
Accuracy			0.97	1200
Macro Avg	0.97	0.97	0.97	1200
Weighted Avg	0.97	0.97	0.97	1200

4.13 Confusion Matrix of KNN Model

A confusion matrix in a K-Nearest Neighbors (KNN) model describes how well the classifier is performing in terms of separating fraudulent transactions from non-fraudulent transactions. This simply comprises the number of true positives (fraudulent transactions that were predicted as fraudulent), true negatives (non-fraudulent transactions that were identified correctly), false positives (non-fraudulent transactions are incorrectly classified as fraudulent), and false negatives (fraudulent transactions that went undetected). The confusion matrix enables as-

sessing the accuracy, precision, recall, and False Positive Rate (misclassifications) of the model to determine how well it performs in identifying fraud as shown in Figure 9. This type of visualization is critical for examining the efficacy of a model on imbalanced datasets, which in fraud detection means that fraudulent transactions represent only a small portion of the overall data.

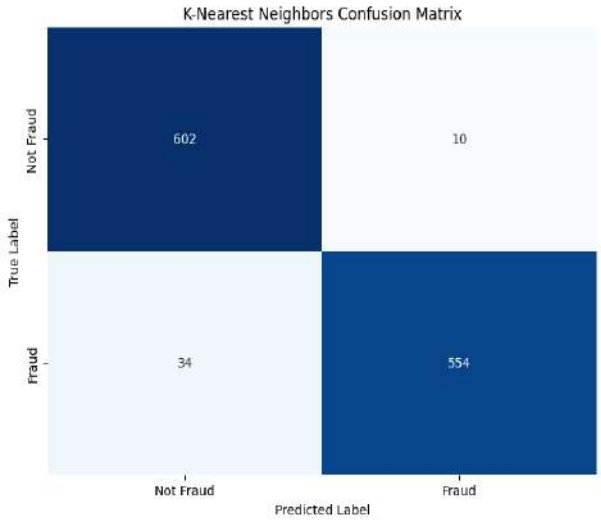

Figure 9. Confusion Matrix of KNN

4.14 Hyperparameter Tuning of XGboost Model

XGB Classifier from the boost library is utilized to construct the XGBoost model. This trains the model with X train and y train thus making predictions on X test. The model performance is then evaluated by calling a classification report to get the details of precision, recall, F1-score, etc. Following training, we make predictions and turn probabilities into binary predictions by using a 0.5 threshold. The XGboost model is assessed with a classification report and an accuracy score as shown in Table 7. These performance metrics are used to compare these models on their fraud detection effectiveness.

Table 7. Classification Report of XGboost Model

Class	Precision	Recall	F1-Score	Support
0	0.96	0.98	0.97	612
1	0.98	0.96	0.97	588
Accuracy			0.97	1200
Macro Avg	0.97	0.97	0.97	1200
Weighted Avg	0.97	0.97	0.97	1200

4.15 Confusion Matrix of XGboost Model

In this section, we use XGBoost for fraud detection and represent it in terms of a confusion matrix. The model is built on the training data (X train and y train). So once training is completed predictions are made on the test dataset (X test) Confusion Matrix is Created, to check the performance of the Model Collected the confusion matrix, is used to determine how well the model has performed in terms of classification of fraudulent and non-fraudulent transactions as shows in Figure 10. A seaborn heatmap-themed annotation of a matrix is used to give

you a quick visual understanding of how the model is doing.

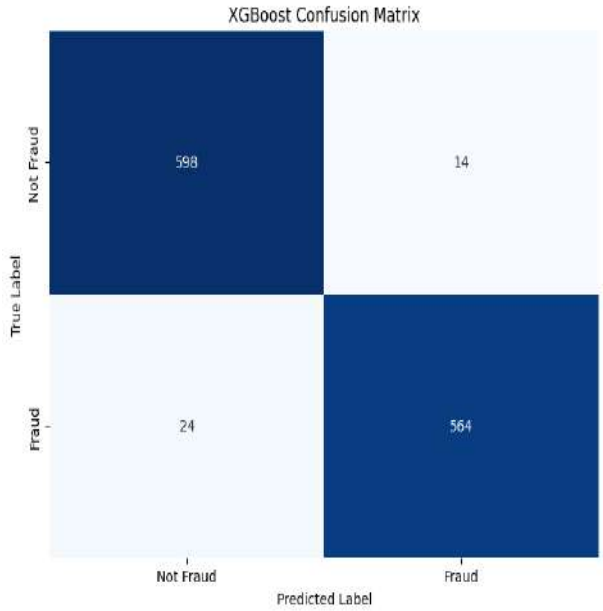

Figure 10. Confusion Matrix of XGboost

4.16 ROC for All Models

We examine the performance of numerous classifiers via an ROC curve. To begin with, we compute the ROC curve for every classifier. The `roc_curve` method from `sklearn.metrics` delivers the False Positive Rate (FPR) and True Positive Rate (TPR) for each model across all predictions. The ROC curve graphs the rates to show how well the model can differentiate positive classes from negative classes.

Models Evaluated:

- Logistic Regression
- K-Nearest Neighbors (KNN)
- SVC (Support Vector Classifier)
- Decision Tree Classifier
- Random Forest Classifier
- XGBoost Classifier

Plotting the ROC Curve: We plot the ROC curves for all the classifiers on the same graph and compare them visually. Each curve depicts the trade-off between sensitivity (true positive rate) and specificity (false positive rate) for every possible threshold. In the legend of each model, the Area Under the Curve (AUC) is also shown for that classifier and as a measure of its general performance. Higher the value of AUC, the models majorly perform well. This visualization allows one to quickly determine which model is likely to be the best fraud detection model by offering a full picture of how each model behaves using different classification thresholds. Figure 11 illustrates the ROC for all models.

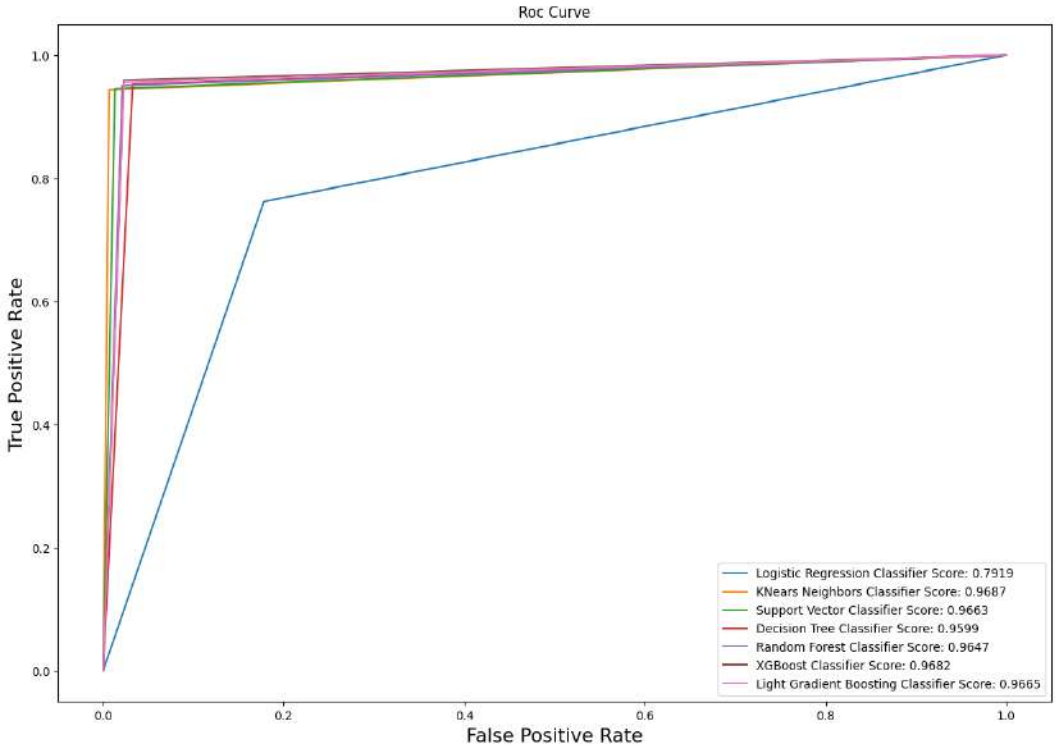

Figure 11. ROC for All Modelst

5 Conclusion

In this research, numerous ML models have been applied to credit card fraud detection which involves LR, SVM, DT, RF, KNN, and XGBoost. Confusion matrices and ROC curves were used to assess the effectiveness of these models. Feature scaling, PCA, and SMOTE were some of the techniques used to improve the performance of models and to oppose the class of imbalance dataset. The study revealed that the models were able to classify fraudulent and non-fraudulent transactions with an accuracy of 96 to 97. Balanced values for precision, recall, and F1-scores of 0.96 to 0.97 were obtained for both classes. The high precision 0.93 to 0.99 and recall 0.92 to 0.99 values confirm that the models could make accurate predictions and be reliable for application. Additional hyperparameter tuning yielded better results in his models too, compounding the reason to be optimizing the model parameters for optimal performance. Therefore, the proposed optimizing ML classifiers for credit card fraud detection on highly imbalanced datasets using PCA and SMOTE techniques creates an essential impact because of the plenty of use of transactions on credit and debit cards in our daily lives. For future work, we suggest incorporating more features or playing around with more advanced algorithms to improve performance. For robustness, we could compare basic models like Logistic Regression to ensemble methods and for detection capabilities, address class imbalance using techniques like SMOTE. It would also be important to evaluate the real-world deployment of our approaches, to judge their performance in practicality.

Author Contributions

Zeeshan Ali Haider: Conceptualization, Methodology, Supervision, Writing—original draft preparation. **Fida Muhammad Khan:** Data curation, Software development, Writing—review and editing. **Abu Zafar:** Investigation, Visualization, Validation. **Nabila:** Resources, Data collection, Project administration. **Inam Ullah Khan:** Writing—review and editing, Validation,editing, Formal analysis.

Compliance with Ethical Standards

It is declare that all authors don't have any conflict of interest.Furthermore, informed consent was obtained from all individual participants included in the study.

References

[1] A. Cherif, A. Badhib, H. Ammar, S. Alshehri, M. Kalkatawi, and A. Imine, "Credit card fraud detection in the era of disruptive technologies: A systematic review," *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 1, pp. 145–174, 2023.

[2] A. A. Khan, A. A. Laghari, H. Elmannai, A. A. Shaikh, S. Bourouis, M. Hadjouni, and R. Alroobaea, "Gan-iotvs: A novel internet of multimedia things-enabled video streaming compression model using gan and fuzzy logic," *IEEE Sensors Journal*, 2023.

[3] A. A. Khan, A. A. Laghari, A. M. Baqasah, R. Alroobaea, A. Almadhor, G. A. Sampedro, and N. Kryvinska, "Blockchain-enabled infrastructural security solution for serverless consortium fog and edge computing," *PeerJ Computer Science*, vol. 10, 2024.

[4] R. Bin Sulaiman, V. Schetinin, and P. Sant, "Review of machine learning approach on credit card fraud detection," *Human-Centric Intelligent Systems*, vol. 2, no. 1, pp. 55–68, 2022.

[5] I. Khan, A. Jameel, I. Ullah, I. Khan, and H. Ullah, "The agi-cybersecurity nexus: Exploring implications and applications," in *Artificial General Intelligence (AGI) Security: Smart Applications and Sustainable Technologies*. Springer, 2024, pp. 271–289.

[6] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms," *IEEE Access*, vol. 10, pp. 39 700–39 715, 2022.

[7] P. Tiwari, S. Mehta, N. Sakhuja, J. Kumar, and A. K. Singh, "Credit card fraud detection using machine learning: a study," *arXiv preprint arXiv:2108.10005*, 2021.

[8] I. Ullah, F. Ali, S. Nazir, H. U. Khan, M. S. Anwar, and C. Choi, "Educating banking employees to ensure security in the cyberworld," in *Cybersecurity Management in Education Technologies*. CRC Press, 2023, pp. 49–63.

[9] E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba, and G. Obaido, "A neural network ensemble with feature engineering for improved credit card fraud detection," *IEEE Access*, vol. 10, pp. 16 400–16 407, 2022.

[10] A. A. Khan, Y.-L. Chen, F. Hajjej, A. A. Shaikh, J. Yang, C. S. Ku, and L. Y. Por, "Digital forensics for the socio-cyber world (df-scw): A novel framework for deepfake multimedia investigation on social media platforms," *Egyptian Informatics Journal*, vol. 27, p. 100502, 2024.

[11] J. I.-Z. Chen and K.-L. Lai, "Deep convolution neural network model for credit-card fraud detection and alert," *Journal of Artificial Intelligence*, vol. 3, no. 02, pp. 101–112, 2021.

[12] A. A. Khan, A. A. Laghari, Z. A. Shaikh, Z. Dacko-Pikiewicz, and S. Kot, "Internet of things (iot) security with blockchain technology: A state-of-the-art review," *IEEE Access*, vol. 10, pp. 122 679–122 695, 2022.

[13] R. Asha and S. K. KR, "Credit card fraud detection using artificial neural network," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 35–41, 2021.

[14] R. Sailusha, V. Gnaneswar, R. Ramesh, and G. R. Rao, "Credit card fraud detection using machine learning," in *2020 4th international conference on intelligent computing and control systems (ICICCS)*. IEEE, 2020, pp. 1264–1270.

[15] A. A. Khan, A. A. Laghari, A. M. Baqasah, R. Alroobaea, T. R. Gadekallu, G. A. Sampedro, and Y. Zhu, "Oran-b5g: A next generation open radio access network architecture with machine learning for beyond 5g in industrial 5.0," *IEEE Transactions on Green Communications and Networking*, 2024.

[16] N. S. Alfaiz and S. M. Fati, "Enhanced credit card fraud detection model using machine learning," *Electronics*, vol. 11, no. 4, p. 662, 2022.

[17] J. F. Roseline, G. Naidu, V. S. Pandi, S. A. alias Rajasree, and N. Mageswari, "Autonomous credit card fraud detection using machine learning approach," *Computers and Electrical Engineering*, vol. 102, p. 108132, 2022.

[18] J. Forough and S. Momtazi, "Ensemble of deep sequential models for credit card fraud detection," *Applied Soft Computing*, vol. 99, p. 106883, 2021.

[19] A. A. Taha and S. J. Malebary, "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine," *IEEE access*, vol. 8, pp. 25 579–25 587, 2020.

[20] H. Fanai and H. Abbasimehr, "A novel combined approach based on deep autoencoder and deep classifiers for credit card fraud detection," *Expert Systems with Applications*, vol. 217, p. 119562, 2023.

[21] Z. Salekshahrezaee, J. L. Leevy, and T. M. Khoshgoftaar, "The effect of feature extraction and data sampling on credit card fraud detection," *Journal of Big Data*, vol. 10, no. 1, p. 6, 2023.

[22] E. Ileberi, Y. Sun, and Z. Wang, "Performance evaluation of machine learning methods for credit card fraud detection using smote and adaboost," *IEEE Access*, vol. 9, pp. 165 286–165 294, 2021.

[23] N. Rtayli and N. Enneya, "Enhanced credit card fraud detection based on svm-recursive feature elimination and hyper-parameters optimization," *Journal of Information Security and Applications*, vol. 55, p. 102596, 2020.

[24] T. T. Nguyen, H. Tahir, M. Abdelrazek, and A. Babar, "Deep learning methods for credit card fraud detection," *arXiv preprint arXiv:2012.03754*, 2020.

[25] F. Wahab, U. Hayat, M. Khan, I. Ullah, and M. Yasir, "Importance of machine learning and network security for communication systems," in *Artificial General Intelligence (AGI) Security: Smart Applications and Sustainable Technologies*. Springer, 2024, pp. 195–210.

[26] A. A. Khan, A. A. Laghari, R. Alroobaea, A. M. Baqasah, M. Alsafyani, R. Bacarra, and J. A. J. Alsayaydeh, "Secure remote sensing data with blockchain distributed ledger technology: A solution for smart cities," *IEEE Access*, 2024.

[27] Y. Lucas and J. Jurgovsky, "Credit card fraud detection using machine learning: A survey," *arXiv preprint arXiv:2010.06479*, 2020.

[28] F. Mehmood, A. A. Khan, H. Wang, S. Karim, U. Khalid, and F. Zhao, "Blpca-ledger: A lightweight plenum consensus protocols for consortium blockchain based on the hyperledger indy," *Computer Standards & Interfaces*, vol. 91, p. 103876, 2025.

[29] I. D. Mienye and Y. Sun, "A deep learning ensemble with data resampling for credit card fraud detection," *IEEE Access*, vol. 11, pp. 30 628–30 638, 2023.

[30] S. Khatri, A. Arora, and A. P. Agrawal, "Supervised machine learning algorithms for credit card fraud detection: a comparison," in *2020 10th international conference on cloud computing, data science & engineering (confluence)*. IEEE, 2020, pp. 680–683.

[31] H. Tingfei, C. Guangquan, and H. Kuihua, "Using variational auto encoding in credit card fraud detection," *IEEE Access*, vol. 8, pp. 149 841–149 853, 2020.

[32] A. A. Khan, S. Dhabi, J. Yang, W. Alhakami, S. Bourouis, and L. Yee, "B-lpoet: A middleware lightweight proof-of-elapsed time (poet) for efficient distributed transaction execution and security on blockchain using multithreading technology," *Computers and Electrical Engineering*, vol. 118, p. 109343, 2024.

[33] P. Chatterjee, D. Das, and D. B. Rawat, “Digital twin for credit card fraud detection: Opportunities, challenges, and fraud detection advancements,” *Future Generation Computer Systems*, 2024.

[34] I. Ullah, A. Noor, S. Nazir, F. Ali, Y. Y. Ghadi, and N. Aslam, “Protecting iot devices from security attacks using effective decision-making strategy of appropriate features,” *The Journal of Supercomputing*, vol. 80, no. 5, pp. 5870–5899, 2024.

[35] I. Benchaji, S. Douzi, B. El Ouahidi, and J. Jaafari, “Enhanced credit card fraud detection based on attention mechanism and lstm deep model,” *Journal of Big Data*, vol. 8, pp. 1–21, 2021.

[36] T.-H. Lin and J.-R. Jiang, “Credit card fraud detection with autoencoder and probabilistic random forest,” *Mathematics*, vol. 9, no. 21, p. 2683, 2021.

[37] S. Bagga, A. Goyal, N. Gupta, and A. Goyal, “Credit card fraud detection using pipeling and ensemble learning,” *Procedia Computer Science*, vol. 173, pp. 104–112, 2020.