Hybrid Machine Learning Approach for Credit Card Fraud Detection: Integrating Supervised and Unsupervised Methods to Enhance Accuracy and Adaptability

P. Anusha[1], B. Santhosh Kumar[2], B. Naganjali[3], Ch. Saketh[4],
G. Akshara[5], D. Sanjana[6], G. Poojitha[7]
[1]Assistant Professor, [2]Professor,[3,4,5,6,7]UG Scholar
Department of CSE, Guru Nanak Institute of Technology, Hyderabad, Telangana, India.
Email id: palagatianushareddy@gmail.com[1], bsanthosh.csegnit@gniindia.org[2],
naganjali0609@gmail.com[3], sakethchary0@gmail.com[4], aksharareddyg05@gmail.com[5],
sanjanachinni0@gmail.com[6], gallanani25@gmail.com[7]

Article Received: 28 April 2025 *Article Accepted: 29 April 2025* *Article Published: 30 April 2025*

Citation
P. Anusha, B. Santhosh Kumar, B. Naganjali, Ch. Saketh,G. Akshara, D. Sanjana, G. Poojitha, "Hybrid Machine Learning Approach for Credit Card Fraud Detection: Integrating Supervised and Unsupervised Methods to Enhance Accuracy and Adaptability", Journal of Next Generation Technology (ISSN: 2583-021X), vol. 5, no. 2, pp. 122-132. April 2025. DOI: 0.5281/zenodo.15666878

Abstract

Machine learning based methodologies for detecting credit card frauds based on the inherent complication of class imbalanced nature of fraud. We start with exploratory data analysis in order to understand some of the feature distributions and relationships, and then apply some feature engineering through binning and encoding to get them into shape for training the model. We employed four different algorithms two of which are supervised (i.e., Logistic Regression and Gaussian Naive Bayes) and the other two are unsupervised anomaly detection methods (i.e, Isolation Forest and Local Outlier Factor). The Logistic Regression algorithm performed best overall but had an even split between false positives and false negative detections of fraud, whilst Naive Bayes provided a cost to the overall fraud detection rate to cover more fraud cases with a high proportion of false positives. Isolation, Forest had higher true positive rates, but poor precision, whereas Local Outlier Factor performed moderately well in detecting fraud. Results indicate supervised methods are more accurate than unsupervised for accurately predicting fraud, and further enhancements can be developed through better preprocessing and ensemble methods.

Keywords: Fraud Detection, Class Imbalance, Anomaly Detection, Supervised Learning,Feature Engineering

I. INTRODUCTION

The quick escalation in and with web transactions has changed economic layouts, granting comfort and efficiency to consumers and establishments alike. Nonetheless, such progression has greatly increased the incidence of credit card scams, creating important economic and security problems globally. Fraudulent activities, often rare yet intense, exploit vulnerabilities inside payment systems, generating huge losses for companies and customers. Discovering certain anomalies is complicated greatly by their scarcity in the middle of legitimate transactions, rendering rule-based methods insufficient. Wide-ranging data-driven understandings have made machine learning a viable answer. It is locating deceit designs via greater precision and pliability. This study deeply examines the use of several machine learning methods. It intends to not ably improve credit card fraud detection, coping with the critical

issue of distinguishing rare fraudulent events inside this imbalanced environment. That study uses a multi-faceted approach, starting from data analysis for finding feature traits, followed through feature engineering within.

II. LITERATURE SURVEY

S. Saini et.al. This paper details machine learning methods used to spot credit card scams, such as the Weighted Product Method to judge multiple models. It looks at metrics and parameters to locate detection ways that are most apt. The limitations of this paper are moving models to changes stays difficult, as models using past data might not shift to new trends. Hostile assaults in which scammers purposely change on details in order to evade spotting from systems. Balancing adequate sensitivity with proper specificity can be difficult. Minimizing false positives might reduce the capability of catching certain fraudulent activities. Unsupervised methods have limited comprehension with transaction behaviour within a context[1]-[3].

Sourav Verma et.al This paper explores into the application of several deep learning algorithms in order for detecting of credit card fraud, targeting for high fraud coverage with truly minimal false positives. It discusses regarding implementing auto-encoders, such as an unsupervised method, for the learning of common transaction patterns. The drawbacks of this paper are the class imbalance problem with extremely few of the fraudulent transactions contrasted to legitimate ones, thus leading to a model bias. There is an inability for detection of merging fraud tactics[1]-[3]. These tactics deviate from historical patterns. Many fraudulent transaction confirmations are delayed considerably, thereby complicating real-time responses. Deep learning models are prone to overfitting, especially since training on limited fraud data.

Ali, Aida et. al This paper is focused on data driven card fraud detection, particularly with the help of machine learning algorithms to automatically detect invalid transactions. It covers the characteristics of datasets, choice of metrics, how to deal with unbalanced datasets, dataset shift (concept drift), approaches to capturing the sequential properties of transactions (feature engineering and sequence modeling methods like recurrent neural networks (LSTM) and hidden Markov models[4]-[6]. The limitations of this paper are difficulty in detecting novel fraud patterns due to the dynamic nature of fraudulent behaviour. Concept drift occurs over time, where transaction patterns change, requiring frequent model updates. High computational costs when using deep learning models for real-time fraud detection. False positives still occur frequently, leading to legitimate transactions being flagged [7]-[8].

III. SCOPE OF THE STUDY

This paper is designed to explore and evaluate machine learning algorithms for detecting credit card fraud in an inherently imbalanced dataset where fraudulent events are rare. Key objectives include, Supervised and unsupervised learning methods for fraudulent prediction. Balancing the need to identify fraud against the need to limit false positives

IV. METHODOLOGY

This study uses a structured method to reach its objectives as follows:

- **Exploratory Data Analysis:** Exploring distributions of features and relationships among them to guide model building
- **Feature Engineering:** The process of making input data informative using binning, encoding, and other techniques.

- **Option of modelling:** Use of Logistic Regression and Gaussian Naive Bayes (move supervised), Isolation Forest and Local Outlier Factor (move unsupervised) to examine different strategies for prediction
- **Evaluation:** Checking how well models can predict, and focusing on how good they are at detecting fraud.
- **Optimization:** One unsupervised model and its hyperparameter tuning for refined Outputs

V. PROBLEM STATEMENT

With rare fraudulent events embedded in significant arrays of legitimate transactions, credit card fraud remains a significant threat to financial security leading to huge losses to the economy as a whole. As a result, traditional techniques for detecting attack patterns are no longer effective due to the imbalance. The suggested solution in this study is in response to the mentioned challenge, where we utilized machine learning in increasing precision in discovering fraud and lowering false positive rates and gained information between supervision and unsupervised strategies to know which approach is efficient in identifying anomaly in an unbalanced environment, and this will ultimately help in prevention in the domain of digital payment systems.

VI. EXISTING SYSTEM

In current credit card fraud detection, financial institutions have generally employed rule-based systems and basic statistical methods to protect against fraudulent transactions. These current systems work by providing certain rules or criteria to follow, like flagging transactions above a particular amount, in strange locations, or that happened in a short amount of time. Get a sample of 10,000 transactions to start with, and check for any if they match these conditions, flag the transaction for a pending investigation, and in most cases, review the transaction manually by fraud analysts.

Moreover, statistical methods, such as outlier detection (based on metrics like mean, standard deviation, or z-scores), can be used to pinpoint anomalies that deviate from normal behaviour. Traditionally, most organizations relied on manually curated rules to identify anomalous network activity, but in recent years, many had begun to integrate simple machine learning models (e.g., decision trees, k-means clustering, or simple supervised classifiers) to complement these rule-based methods with data-driven insights.

Despite their pervasive usage, present systems contain key defects, especially when meeting the challenge of seeing few fraudulent behaviours within a wide group of real trades. Rule- based methods remain always static, relying upon defined thresholds or patterns derived from prior fraud cases. Such inflexibility severely curtails their skill to shift to new fraud methods. This inflexibility often creates unseen bogus events (false negatives) in actuality. The sluggishness weighs at fraud spotting groups with undue hand reviews, grows working prices of them, and angers patrons with routine deal breaks from this. The initial machine learning uses, although advances, seem basic, frequently failing with biased data. For instance, unsupervised clustering may group anomalies together but struggles to distinguish between fraudulent activities from and legitimate outliers, such as large purchases and/or travel-related spending. Supervised models, conversely, depend deeply on labeled fraud data, which is scarce as well as costly to obtain, limiting their scalability along with generalization. Consequently, most existing systems exhibit

suboptimal performance regarding precision as well as recall, failing to provide a strong solution for modern fraud detection needs wherein adaptability and accuracy are primary.

VII. PROPOSED SYSTEM

The recent research aims to overcome the drawbacks of conventional fraud detection techniques by introducing an improved machine learning or AI-based approach. This is an integrated system of both supervised and unsupervised techniques, focused on improving anomaly identification and adaptability to unbalanced surroundings. Compared to static rule-based approaches, it is more flexible, scalable, and generates fewer false positives.

1. **Data set Explanation:**
   The dataset contains 284,807 credit card transactions used for fraud detection. It includes 30numerical features (V1–V28) obtained via PCA, along with Time, Amount, and Class labels. The Time column represents transaction timing, and Amount shows the transaction value. The Class label (0 = Normal, 1 = Fraud) identifies fraudulent transactions. This dataset helps in detecting anomalies in banking systems. It can be integrated into your ATM Simulator System for real-time fraud detection.
2. **Data Preparation:**
   Proposed structure begins from a sound data management phase for guaranteeing algorithms efficiently perceive fraudulent-type actions.
3. **Exploratory Data Analysis:** The step thoroughly probes features & traits in addition to relationships, as well as distributions. It is furnishing a thorough data-led basis into modelling by detection of faint signals in conduct which are bogus.
4. **Feature Engineering:** Raw inputs are transformed via methods like binning continuous variables, plus encoding, which optimizes data for ml.
5. **Model Implementation:** At its core, the system deploys in parallel four different algorithms to take advantage of labelled and unlabeled data
6. **Supervised Methods:** Functions like Logistic Regression and Gaussian Naive Bayes that leverage the available labels. While Logistic Regression takes a probabilistic approach to consider feature weights, Naive Bayes technique uses a quick approach in which it calculates the independency of the features for faster pattern matching.
7. **Unsupervised Methods:** Isolation Forest and Local Outlier Factor to detect anomalies. Local Outlier Factor (LOF)5 computes a score to identify the outliers and Local outlier factors assesses local density variations to flag suspicious events
8. **Evaluation Process:** Models are subjected to strict evaluations to guarantee that they appropriately detect fraud: Data
9. **Splits:** For training and testing models on separate data splits, the system uses detected frauds but also checks for false positives
10. **Performance Assessment**: Methods & measures like accuracy, precision, recall, and visual tools such as ROC curves and confusion matrices give a detailed analysis of each method's strengths.

A. Optimization

Hyperparameter Tuning:

If used with Isolation Forest, parameters get optimized so that fraud detection goals will align, giving flexibility that static systems lack. The proposed system seeks to greatly outperform standard methods in many ways through accurately identifying fraud It also greatly reduces from false positives, as well as providing on a fully scalable platform. By blending together supervised and unsupervised ways, it gives a flexible answer, readying it for future improvements such as finer preprocessing or ensemble ways to increase financial security.

VIII. SYSTEM ARCHITECTURE

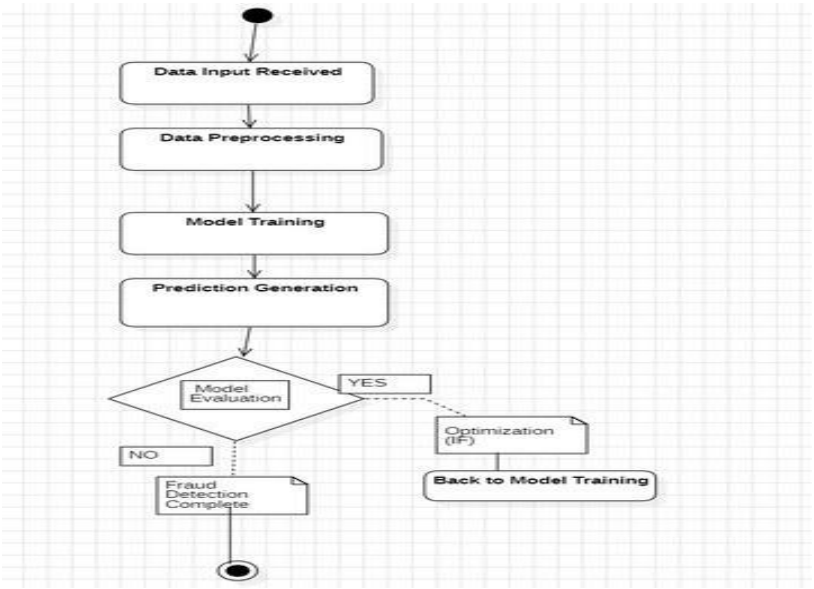

Fig 1: Flowchart for fraud Detection

This flowchart represents a fraud detection using machine learning.

1. Data Input Received:
The system collects raw data, which include transaction logs, user behaviors, timestamps, IP addresses etc.

2. Data Preprocessing
This step involves cleaning and transforming data to ensure it is suitable for model training.

- Handling Missing Values
- Feature Engineering
- Data Normalization/Scaling

3. Model Training
A machine learning model is trained using historical data. Algorithms such as Decision Trees, Random Forest, Support Vector Machines (SVM), Neural Networks, or Deep Learning (LSTMs, CNNs, etc.) can be used.

4. Prediction Generation
Once trained, the model predicts whether a new transaction is fraudulent or not. The output could be, a binary classification (Fraud / Not Fraud). A fraud score (Probability of fraud).

5. Model Evaluation
The system evaluates the accuracy of predictions using metrics such as:

- **Precision &amp:** Recall: To measure fraud detection effectiveness.
- **F1-score:** A balance between precision and recall.
- **Confusion Matrix:** Analysis how many fraud cases were correctly or incorrectly classified.

6. Decision Process

If the model performs well (passes accuracy thresholds), fraud detection is complete. If the model performs poorly, it undergoes optimization.

7. Optimization (If Required):

- Hyperparameter tuning: Adjusting parameters like learning rate, batch size, number of hidden layers, etc.
- Feature selection: Adding or removing features that impact fraud detection.
- Algorithm change: Switching to a more advanced algorithm if needed.

8. Back to Model Training (If Optimization is Done)

The optimized model is trained again with better features or tuned hyperparameters. This loop continues until the model performs optimally.

9. Fraud Detection Complete

Once the model meets the required accuracy, it is deployed for real-time fraud detection in financial transactions, e-commerce, or cyber security applications.

IX. ALGORITHM

SUPEVISED ALGORITHM (LOGISTIC REGRESSION AND NAVIES BAYES ALGORITHM) UNSUPERVISED ALGORITHM(LOCAL OUTLIER FACTOR,ISOLATION FOREST)

1. Data Preprocessing and Exploration

FUNCTION Preprocess And Explore Data(dataset)
Table Load Dataset(dataset)
backup ← Copy(table)
DISPLAY stats
classCounts ← CountOccurrences(table[Class&]) // 0: non-fraud, 1: fraud DISPLAY classCounts
VISUALIZE BarChart(classCounts, x=Class y=Count&, title=Fraud vs Non-Fraud
RETURN table END FUNCTION

2. Calculate Outlier Fraction

FUNCTION Calculate Outlier Fraction(table)
shuffled TableShuffle(table, randomSeed)
fraudSubset ← Filter(shuffledTable, Class = 1) validSubset ← Filter(shuffledTable, Class= 0) outlierFraction ← Size(fraudSubset) / Size(validSubset) DISPLAY "Outlier Fraction: ",outlierFraction
fraudStats ← ComputeStats(fraudSubset[Amount&]) validStats ← ComputeStats(validSubset[Amount]) DISPLAY "Fraud Amount Stats: ",fraudStats
DISPLAY "Valid Amount Stats: ",validStats

3. Train-Test Split

FUNCTION SplitTrainTest(table)
X ← DropColumn(table, Class // Features Y ← table[Class]// Target

X_train, X_test, Y_train, Y_test ← SplitData(X, Y, testSize=0.3, randomSeed, shuffle=True)
DISPLAY Shapes(X_train, X_test, Y_train, Y_test)

4. Isolation Forest Anomaly Detection

FUNCTION IsolationForestModel(X_train, X_test, outlierFraction, randomSeed)
Model InitializeIsolationForest(nTrees=100, contamination=outlierFraction, seed=randomSeed)
Train(model, X_train) // Build isolation trees
predictions ← Predict(model, X_test) // 1: inlier, -1: outlier
binaryPreds ← ConvertPredictions(predictions) // 1 → 0 (non-fraud), -1 → 1 (fraud)

5. Local Outlier Factor (LOF) Anomaly Detection

FUNCTION LocalOutlierFactorModel(X_test, outlierFraction)
model InitializeLOF(nNeighbors=20, contamination=outlierFraction)
predictions ← FitPredict(model, X_test) // 1: inlier, -1: outlier
binaryPreds ← ConvertPredictions(predictions) // 1 → 0 (non-fraud), -1 → 1 (fraud)

6. Model Evaluation

FUNCTION EvaluateModel(Y_true, Y_pred, modelName)
Accuracy ComputeAccuracy(Y_true, Y_pred)
DISPLAY modelName, " Accuracy: ", accuracy
report ← GenerateClassificationReport(Y_true, Y_pred, classes=[Valid&, & Fraud&])
DISPLAY report
cm ← ComputeConfusionMatrix(Y_true, Y_pred) // TN, FP, FN, TP VISUALIZE ConfusionMatrixHeatmap(cm, labels=[Valid &,Fraud], title=modelName + " Confusion Matrix&quot 😉

7. Model Comparison

FUNCTION CompareModels(metricsIso, metricsLOF)
Table CreateTable(columns=[Classifier,Accuracy,Precision (Fraud),Recall (Fraud),F1-Score (Fraud),False Positives,False Negatives])
AddRow(table, Isolation Forest,metricsIso) AddRow(table, Local Outlier Factor, metricsLOF)
DISPLAY "Comparison of Outlier Detection Algorithms: ", table VISUALIZE

8. Hyperparameter Tuning

FUNCTION TuneHyperparameters(X_train, modelType)
paramSpaceDefineParameters(modelType) // e.g., nTrees, contamination
bestParams ← RandomizedSearch(paramSpace, nIterations=10, cv=3, scoring=, seed=randomSeed)
tunedModelInitializeModel(modelType, bestParams)
Train(tunedModel, X_train)

9. Improve model performance

X_train_resampled, Y_train_resampled ← OversampleMinority(X_train, Y_train)
tunedIsoModel ← TuneHyperparameters(X_train_resampled, "IsolationForest&quot
X_enhanced, Y_enhanced ← EnhanceFeatures(table)
X_train_enh, X_test_enh, Y_train_enh, Y_test_enh ← SplitTrainTest(X_enhanced)

10. Retrain and evaluate tuned model

tunedPreds ← Predict(tunedIsoModel, X_test_enh)

tunedMetrics ← EvaluateModel(Y_test_enh, tunedPreds, "Tuned Isolation Forest&quot

X. RESULTS AND DISCUSSION

```
   Time        V1        V2        V3        V4        V5        V6        V
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.23959
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.07880
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.79146
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.23760
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.59294

         V8        V9  ...       V21       V22       V23       V24       V25
0  0.098698  0.363787  ... -0.018307  0.277838 -0.110474  0.066928  0.128539
1  0.085102 -0.255425  ... -0.225775 -0.638672  0.101288 -0.339846  0.167170
2  0.247676 -1.514654  ...  0.247998  0.771679  0.909412 -0.689281 -0.327642
3  0.377436 -1.387024  ... -0.108300  0.005274 -0.190321 -1.175575  0.647376
4 -0.270533  0.817739  ... -0.009431  0.798278 -0.137458  0.141267 -0.206010

        V26       V27       V28  Amount  Class
0 -0.189115  0.133558 -0.021053  149.62      0
1  0.125895 -0.008983  0.014724    2.69      0
2 -0.139097 -0.055353 -0.059752  378.66      0
3 -0.221929  0.062723  0.061458  123.50      0
4  0.502292  0.219422  0.215153   69.99      0

[5 rows x 31 columns]
```

Fig 2: Data set of the credit card Transactions

The [5 rows x 31 columns] at the bottom indicates that there are 5 rows and 31 columns in the dataset. A single credit card transaction is represented by each row, and a feature of that transaction is represented by each column. The Class column in the dataset, which is used for fraud detection, specifies whether a transaction is fraudulent (1) or not (0).

	Time	V1	V2	V3	V4	V5	V6
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05
mean	94813.859575	1.168375e-15	3.416908e-16	-1.379537e-15	2.074095e-15	9.604066e-16	1.487313e-15
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01

8 rows × 31 columns

FIG 3: Statistical summary of the data set

Using Pandas describe() method, the image displays a statistical summary of a credit card transaction dataset (`creditcard.csv`) containing 284,807 rows and 31 column `Time`, `V1` to`V28` (PCA-transformed), `Amount`, and Class (0 for non-fraud, 1 for fraud) are some of its features. `count`, `mean`, `std`, `min`, `25%`, `50%`, `75%`, and `max` are important statistics. High imbalance (0.1727% fraud) is confirmed by the Class mean (`0.001727`). {Amount` shows skewness, ranging from `0.0` to `25691.16` (mean {88.349619`, median {22.0`). For PCA features, the means of `V1` to `V28` are close to 0 and the standard

deviation is approximately 1.Outliers are suggested by the large range in the `V` columns (e.g., `V1` from `-56.40751` to`2.45493`).

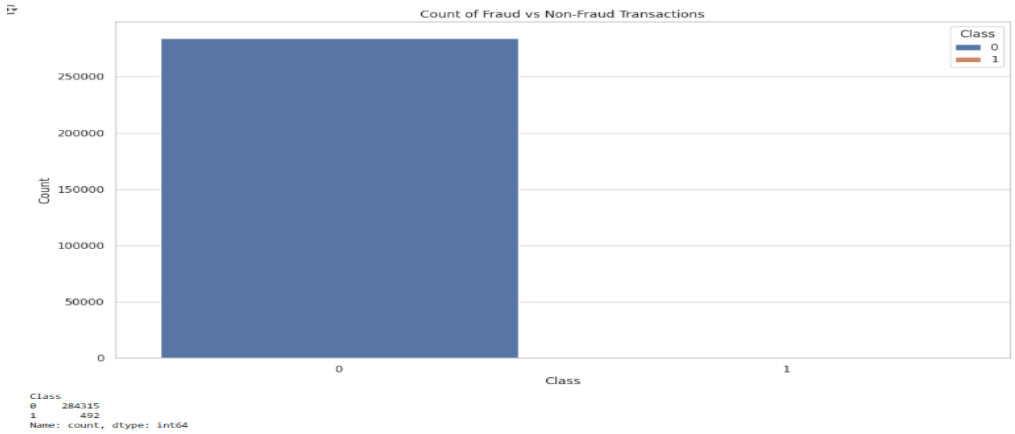

Fig 4: Bar Graph of fraud and non-fraud transactions

The dataset distribution of non-fraud (Class 0) and fraud (Class 1) transactions is shown in the bar chart. With the label "Class," the x-axis shows the two types of transactions: fraudulent and non-fraudulent. With the label "Count," the y-axis displays the quantity of transactions in each category. A clear class imbalance is highlighted by the visualization, which shows that non-fraudulent transactions (Class 0) much outnumber fraudulent ones (Class 1). The two classes are distinguished by the legend in the upper right corner. A fraud detection model performance maybe impacted by this mismatch, which could skew it in favour of the majority class. Techniques like resampling, creating synthetic data, or cost-sensitive learning might be required to increase accuracy.

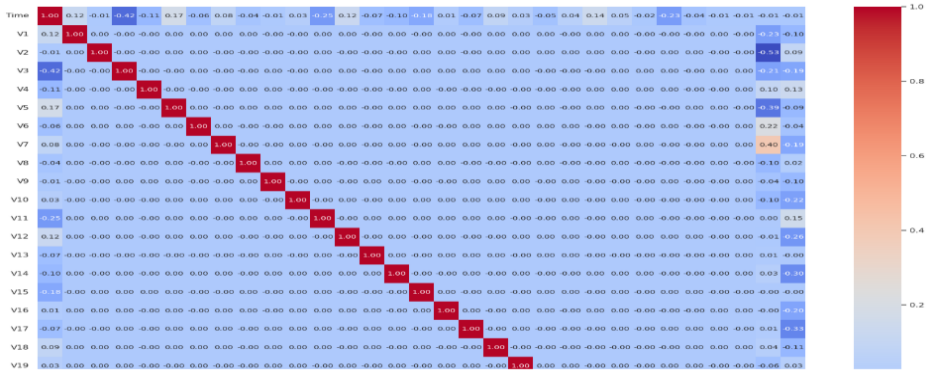

Fig 5: Correlation Matrix of Features in Fraud Detection Dataset through heatmap

The association between the dataset various properties is displayed visually in the heat map. The feature names, such as Time, V1, V2, V3, etc., are represented by the X-axis and Y-axis. A correlation coefficient, which ranges from -1 to 1, is present in every cell and indicates the direction and intensity of correlations between features. Since all of the diagonal values are 1,each feature has a perfect correlation with itself. The degree of correlation is displayed by the color bar (legend) on the right, where blue " denotes a negative correlation and red & quot; indicates a strong positive correlation. Variables with lighter hues have little to no link with one another. Dependencies between features may be

shown by strong correlations, which could affect the model performance. Multicollinearity caused by highly correlated characteristics necessitates feature selection strategies.

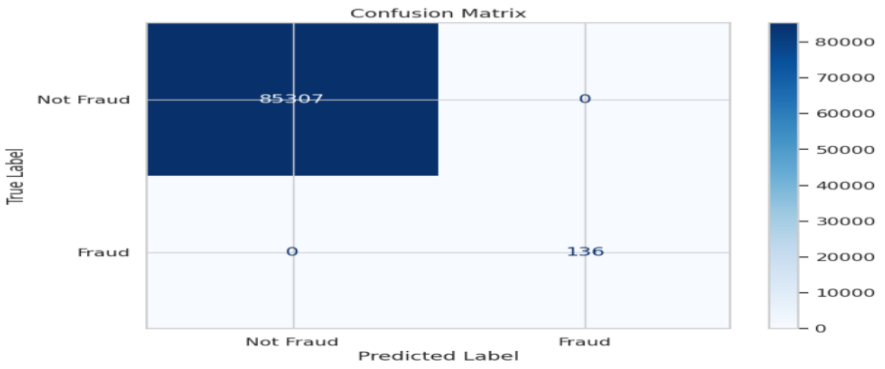

Fig 6: Confusion Matrix for Fraud Detection Model

The confusion matrix shows how well the fraud detection model performs. True Positives, True Negatives, False Positives, and False Negatives are its four main values. 85,307 transactions is this instance were appropriately categorized as "NotFraud," whereas 136 transactions were appropriately labeled as "Fraud." No cases have been incorrectly classified, resulting in neither false positives nor false negatives. This suggests an error-free, extremely precise model. Each classification count is represented by the color intensity. While off-diagonal numbers signify misclassifications, diagonal values display accurate predictions. This confusion matrix indicates that the model detects fraud very well.

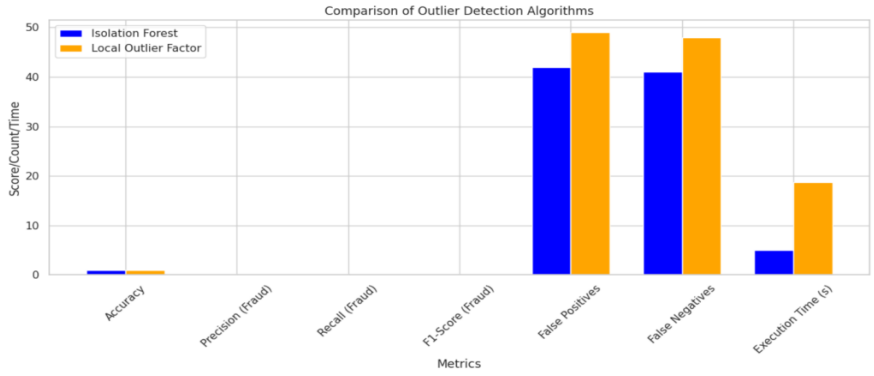

Fig 7: Comparison of Outlier Detection Algorithms for Fraud Detection

Isolation Forest and Local Outlier Factor are two well-known outlier detection algorithms whose performances are contrasted in this bar chart. Accuracy, precision, recall, F1-score, false positives, false negatives, and execution time are among the parameters that form the basis of the evaluation. While the accuracy of both algorithms is comparable, Isolation Forest exhibits marginally superior precision and recall. However, a higher error rate in categorization is shown by the Local Outlier Factor higher number of false positives and false negatives. In comparison to Isolation Forest, Local Outlier Factor has a substantially longer execution time and is therefore computationally more costly. These findings imply that Isolation Forest would be a better option for fraud detection since it offers faster execution, decreased false classifications, and improved precision and recall.

XI. CONCLUSION

The advantages and disadvantages of Isolation Forest and Local Outlier Factor for fraud detection are contrasted. With a shorter execution time and comparatively fewer false positives and false negatives, Isolation Forest operated effectively. The Local Outlier Factor, on the other hand, discovered more outliers and had a higher false positive rate despite having a slightly longer execution time. Both approaches are capable of detecting fraudulent transactions, but how well they work relies on the dataset and the acceptable balance between recall and precision. According to the analysis, Local Outlier Factor is helpful when identifying even small anomalies is crucial, while Isolation Forest is better for quicker anomaly detection with middling accuracy.

REFERENCES

[1]. S. Saini, G. Bathla and V. Wasson, "Credit Card Fraud Detection using Machine Learning: Methods, Challenges and Solutions," *2024 International BIT Conference (BITCON)*, Dhanbad, India, 2024, pp. 1-6, doi: 10.1109/BITCON63716.2024.10985587.

[2]. S. A. Adewumi and A. A. Akinyelu, "A survey of machine-learning and nature-inspired based credit card fraud detection techniques," *Int. J. Syst. Assur. Eng. Manag.*, vol. 11, no. 3, pp. 527–546, Jun. 2020.

[3]. R. D. Sudharsan and D. M. Lyun, "Detecting credit card fraud by decision trees and support vector machines," in Proc. Int. MultiConf. Eng. Comput. Scientists (IMECS), Mar. 2011, pp. 442–447.

[4]. Ali, Aida & Shamsuddin, Siti Mariyam & Ralescu, Anca. (2015). Classification with class imbalance problem: A review. 7. 176-204.

[5]. Zhu, Shunzhi & Ma, Ying & PAN, Weiwei & Zhu, Xiatian & LUO, Guangchun. (2014). Balanced Neighborhood Classifiers for Imbalanced Data Sets. IEICE Transactions on Information and Systems. E97.D. 3226-3229. 10.1587/transinf.2014EDL8064.

[6]. Río, Sara & López, Victoria & Benítez, José & Herrera, Francisco. (2014). On the use of MapReduce for Imbalanced Big Data using Random Forest. Information Sciences. 285. 112-137. 10.1016/j.ins.2014.03.043.

[7]. Praveen, S. P., Vellela, S. S., & Balamanigandan, R. (2024). SmartIris ML: harnessing machine learning for enhanced multi-biometric authentication. Journal of Next Generation Technology (ISSN: 2583-021X), 4(1).

[8]. Dr. Ranga Swamy Sirisati, A. Kalyani, V. Rupa, Dr.Pradeep Venuthurumilli, Md Ameer Raza (2024). "Recognition of Counterfeit Profiles on Communal Media using Machine Learning Artificial Neural Networks & Support Vector Machine Algorithms", Journal of Next Generation Technology (ISSN: 2583-021X), 4(2), pp. 19-27. May 2024.