International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Comparative Analysis of Machine Learning Techniques for Detecting Credit Card Fraud

Rajani P. K.*[1], Roshan Mathew[2], Atharva Walawalkar[3], Prathmesh Patil[4], Ujwal Shirode[5] , Ajjay Gaadhe[6]

Submitted: 26/01/2024 **Revised**: 04/03/2024 **Accepted**: 12/03/2024

Abstract: Fraudulent use of credit cards is a major problem across the world, causing enormous financial losses for banks, retailers, and customers. Machine learning algorithms are effective in detecting fraudulent transactions, but the imbalanced dataset with the majority of transactions being legitimate poses a challenge. The SMOTE technique is used to address the imbalanced dataset caused by the minority class (fraudulent transactions) in this study, which assesses the effectiveness of multiple machines learning classifiers, including Logistic Regression, Random Forest, Decision Tree, Gradient Boosting, XGBoost, as well as SVM. The Random Forest model is the most effective classification algorithm overall, with a recall of 0.8482, precision of 0.8526 and F1 score of 0.8504 in the 60:40 split. It had a recall of 0.8603, precision of 0.8357 and F1 score of 0.8478 for the 70:30 split. For the 80:20 split, it had a recall of 0.8367, precision of 0.9213 and F1 score of 0.8770. The study indicates that the SMOTE approach and various classifiers are effective for detecting credit card fraud, with Random Forest being the best-performing classifier.

Keywords: *Decision Tree, Gradient Boosting, Random Forest, SMOTE, SVM, XGBoost*

1. Introduction

Fraudulent use of credit cards is a serious concern for financial organizations as it results in substantial financial losses and undermines consumer confidence in the banking system. Machine learning techniques have shown promise in detecting fraudulent transactions by analyzing historical data patterns. However, the issue of imbalanced datasets, where fraudulent transactions comprise only a small proportion of legitimate transactions, makes effective detection of fraud challenging. One approach that has been proposed to overcome this challenge is the Synthetic Minority Oversampling Technique (SMOTE).

This paper explores the use of SMOTE in the detection of credit card fraud to address the issue of imbalanced datasets and assesses the effectiveness of various classifiers. The study makes use of a Kaggle dataset which comprises data from European cards collected over a two-day period in the month of September 2013. The study's data is extremely skewed, with just 492 instances of fraud out of 2,84,315 transactions. Only 0.172% of the entire transactions are classified as positive (frauds). The dataset solely comprises numerical input variables resulting from PCA transformation, with variables V1 through V28 serving as the principal components. The only variables not modified by PCA are the features: 'Time' and 'Amount.' The 'Time' feature provides the seconds that elapsed between every single transaction and the initial transaction in the dataset, while the 'Amount' feature contains the transaction worth. The response for the variable, labelled 'Class,' has a value of 1 in the event of fraud and 0, in the absence of it.

Prior studies have used various techniques to deal with the imbalance in data, including one-class classification, random oversampling, and cost-sensitive models [1]. To build the classification model for the detection of credit card fraud, most studies employed classifiers such as Random Forest, Decision Tree, SVM, and KNN ([1]-[6]). Some studies have also used deep learning techniques to build predictive models ([7], [8], [11], [13]).

This study proposes using the SMOTE oversampling technique to address the issue of imbalanced datasets and fit various classifiers to predict fraudulent transactions. Each classifier's performance is evaluated using multiple types of metrics, including Accuracy, Recall, F1 Score, Precision, Area Under Curve Receiver Operating Characteristics Score (AUC-ROC Score), as well as Matthews Correlation Coefficient (MCC).

[1,5,6] *Faculty of Electronics and Telecommunication Department, Pimpri Chinchwad College of Engineering, Pune – 411044, INDIA*
[1] *ORCID ID: 0000-0003-3635-2094*
[2] *Department of Electronics and Telecommunication Engineering, Pimpri Chinchwad College of Engineering, Pune – 411044, INDIA*
ORCID ID: 0009-0005-1196-597X
[3] *Department of Electronics and Telecommunication Engineering, Pimpri Chinchwad College of Engineering, Pune – 411044, INDIA*
ORCID ID: 0009-0004-5478-520X
[4] *Department of Electronics and Telecommunication Engineering, Pimpri Chinchwad College of Engineering, Pune – 411044, INDIA*
ORCID ID: 0009-0002-2354-066X
[5] *ORCID ID: 0000-0001-7644-0991*
[6]*ORCID ID: 0000-0001-6728-5264*
* *Corresponding Author Email: rajani.pk@pccoepune.org*

2. Methodology

2.1. Data Preparation

The Kaggle detection of credit card fraud data was imported into a Python notebook, and a check for null values was performed, revealing no null values in the data. To visualize the distribution between fraudulent and non-fraudulent transactions, a count plot was created.

The count plot shown below (Fig. 1.) illustrates a significant imbalance in the dataset, with non-fraudulent transactions (0) having a count of 2,84,315 greatly outnumbering fraudulent transactions (1) having a count of 492. Subsequently, a correlation plot was plotted to explore the relationship between the "Class" label and the other features.

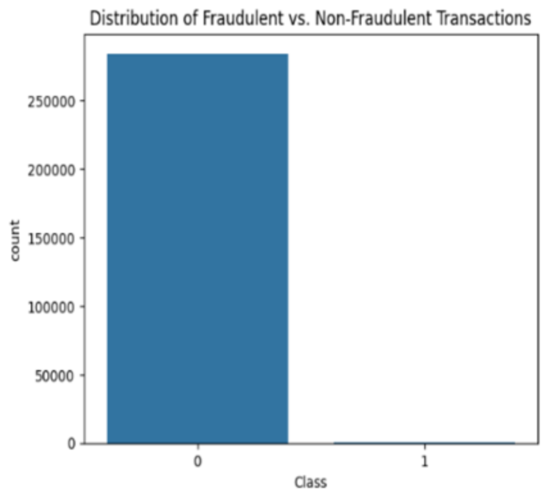


Fig. 1. Count plot depicting the distribution of fraudulent transactions vs. non-fraudulent transactions

2.2. Synthetic Minority Over-sampling Technique (SMOTE)

In classification problems, imbalanced datasets occur when the amount of samples in a particular class is much lower than the number of samples in the other class, resulting in poor performance of the models, particularly for the class which is in the minority. To address this issue, SMOTE (Synthetic Minority Over-sampling Technique) is used which is a prominent data augmentation approach. SMOTE generates synthetic minority class samples through interpolation. The algorithm selects a sample from the class which is in the minority, identifies its k nearest neighbors, and creates new synthetic samples by interpolating between the chosen sample and its neighbors. This approach oversamples the minority population, balancing the amount of specimens in each of the classes and improving the quality of the minority class samples. In this study, the dataset for the detection of credit card fraud exhibits a significant class imbalance, with 2,84,315 non-fraudulent transactions (majority class) and only 492 fraudulent transactions (minority class). Therefore, SMOTE is utilized to synthesize new minority class samples and balance the class distribution.

2.3. Train-Test Split of Data

The efficacy of various machine learning classifiers for identifying credit card fraud is investigated in this work utilizing three different train-test data splits: 60:40, 70:30, and 80:20. The primary goal of using different splits is to evaluate the effect of modifying the size of data utilized in training and testing upon the performance of classifiers.

To ensure the reproducibility and comparability of results, a fixed random seed of 42 is set when splitting the data. This practice guarantees that each execution of the code generates the same split of data, facilitating a fair comparison of the classifiers' performance.

To avoid any bias during the training and testing processes, feature scaling is performed on the dataset after it has been divided into sets for training and testing. Feature scaling normalizes the data and ensures that each feature has the same scale, making it easier for many machine-learning algorithms to function optimally.

The data in this study is scaled to have a mean of zero and a standard deviation of one using standard scaling. This technique transforms the data into a more appropriate form for the classifiers to learn from and minimizes the impact of outliers or extreme values in the data.

2.4. Algorithms

2.4.1. Logistic Regression

A statistical technique employed for modeling the relationship between a dependent variable that is categorical and independent variables is known as logistic regression. It assesses the likelihood of the occurrence of an event by using a logistic function, although significant biases might come up if the conditions of linearity between the variables that are independent and the variable that is dependent are not fulfilled. The logistic function's formula is as follows:

$$p(x) = \frac{1}{1+e^{-z}} \tag{1}$$

where p(x) is the likelihood of the event occurring given the independent variable values and z is the linear combination of the variables that are independent.

2.4.2. Decision Tree

The decision tree is a widely employed machine learning technique for classification and regression tasks, offering a comprehensible representation of decision-making processes. It maximizes class separation or minimizes regression error by repeatedly partitioning input data based on informative features, making it valuable for interpretable and accurate predictions. The entropy is given by:

$$E(S) = -p(+)logp(+) - p(-)logp(-) \tag{2}$$

where p(+) is the likelihood of the class that is positive, p(-) is the likelihood of the class that is negative, and S is the training sample subset.

The information gain is as follows:

$$Information\ Gain = E(Parent) - E(Parent|Child) \quad (3)$$

2.4.3. Random Forest

Random Forest, an ensemble learning algorithm, is widely employed for diverse predictive modeling applications such as classification and regression. By aggregating outputs from multiple decision trees, each constructed on random subsets of data and features, it effectively mitigates overfitting, enhances generalization, and offers advantages such as robustness to noise, high-dimensional data handling, and feature importance estimation with minimal hyperparameter tuning. The formula for Random Forest is given by:

$$f(x) = \frac{1}{T}\sum_{1}^{T} ft(x) \quad (4)$$

where ft(x) is the prediction of the tth tree, T represents the total amount of trees, and 1/T represents the normalization factor.

2.4.4. Gradient Boosting

Gradient Boosting, a potent ensemble learning algorithm, is utilized for regression and classification tasks, effectively combining weak models to construct a robust predictive model. By iteratively minimizing residual errors through negative gradient fitting, it addresses complex relationships, missing data, and feature importance estimation, yielding accurate predictions with limited data and reducing overfitting concerns. The formula for Gradient Boosting is given by:

$$Fm(x) = Fm - 1(x) + \gamma mhm(x) \quad (5)$$

where m is the amount of decision trees made, Fm-1(x) is the base model's prediction (previous prediction), γm is the multiplicative factor and hm(x) is the recent decision tree made on the residuals.

2.4.5. XGBoost

XGBoost, an influential machine learning algorithm, finds widespread application in supervised learning tasks like regression, classification, and ranking. Utilizing decision trees as base models, XGBoost excels in delivering remarkably accurate results while maintaining computational efficiency. The formula for XGBoost is given by:

$$Fm(x) = Fm - 1(x) + \gamma mhm(x) \quad (6)$$

where m is the amount of decision trees made, Fm-1(x) is the base model's prediction (previous prediction), γm is the multiplicative factor and hm(x) is the recent decision tree made on the residuals.

2.4.6. Support Vector Machine (SVM)

Support Vector Machine (SVM), a popular and powerful machine learning technique, is extensively applied in supervised tasks encompassing classification, regression, and outlier detection. SVM's notable strength lies in effectively managing high-dimensional data and nonlinear relationships by utilizing kernel functions to map data to a higher-dimensional space, facilitating optimal hyperplane identification for class separation. The formula for SVM can be represented as:

$$f(x) = sgn(w^T x + b) \quad (7)$$

where w represents the weight vector, b represents the bias term, and sgn represents the signum function, which returns either +1 or -1 based on the sign of the argument.

2.5. Workflow

The first step involved in building the machine learning model, depicted in the figure (Fig. 2.), is data pre-processing. It includes several tasks such as identifying the count of fraudulent and non-fraudulent transactions, checking for any missing or null values, and analyzing the correlation between the features and the class label. A correlation plot can be plotted to visualize the correlation matrix and identify which features have no significant impact on the class label. These inconsequential features can be removed from the dataset to minimize its dimensionality.

The dataset is then divided into training and testing data. In this paper, three different splits have been implemented, 60:40, 70:30, and 80:20. The training dataset is employed for training the model, while the testing dataset is employed to assess the model's performance.

The data is scaled before the model is trained to ensure that all features are on the same scale. This helps the model to understand the trends in the data and increases its performance.

The dataset used is imbalanced, with a high proportion of non-fraudulent transactions and a low proportion of fraudulent transactions. This can lead to biased models, where the classifier tends to predict the majority class. To tackle this issue, the Synthetic Minority Over-sampling Technique (SMOTE) is employed to generate synthetic data and oversample the class which is in the minority. This technique helps to balance the dataset and improves the performance of the classifier.

Using the dataset for training, various classifiers like Logistic Regression, Random Forest, Decision Tree, XGBoost, Gradient Boosting, and Support Vector Machine (SVM) are taught. The model is assessed after training using the testing dataset and various metrics such as precision, accuracy, F1 score, recall, Matthews Correlation Coefficient (MCC), and AUC ROC score. These metrics help in measuring the model's performance in identifying transactions that are fraudulent.

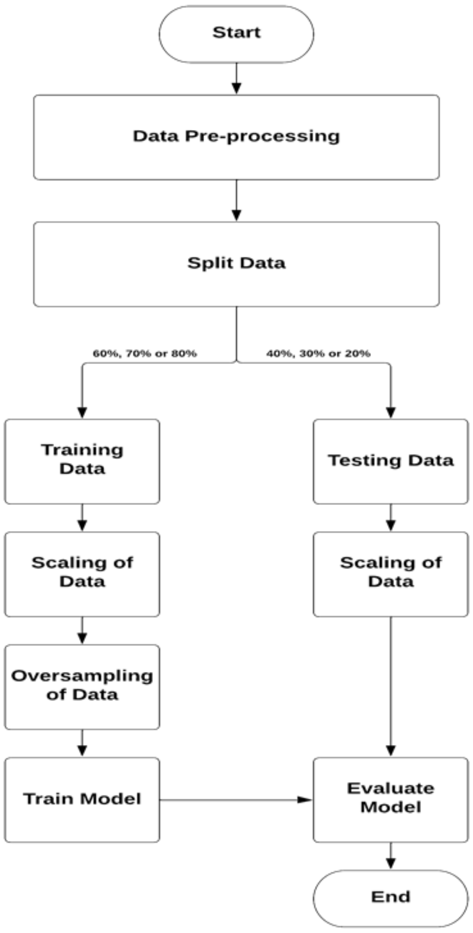


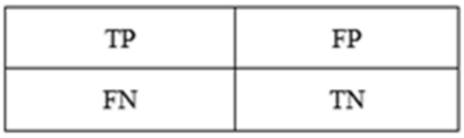
Fig. 2. Flowchart depicting the machine learning model's workflow

3. Results and Discussion

3.1. Evaluation Metrics

3.1.1. Confusion Matrix

The confusion matrix, a fundamental tool in assessing the performance of classification models, presents a tabular summary of correct and incorrect predictions on test data. By partitioning predictions into True Positives, True Negatives, False Positives, and False Negatives, it enables the evaluation of crucial metrics such as precision, accuracy, F1-score, and recall, thus offering valuable insights for research purposes.



TP	FP
FN	TN

Fig. 3. Binary Classification Confusion Matrix

The figure shown above (Fig. 3.) depicts a typical binary classification confusion matrix which gives an idea about the count of TP, FP, TN, and FN.

3.1.2. Accuracy

Accuracy, a commonly employed metric in the area of classification, quantifies the percentage of properly classified instances within a dataset, reflecting the model's proficiency in assigning the correct class labels. However, this metric exhibits limitations by not accounting for the varying significance of different types of errors and proving inadequate for imbalanced datasets, where alternative evaluation metrics like recall, precision, area under the ROC curve, F1-score, etc., play a crucial role in comprehensive model assessment ([12], [14]).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

3.1.3. Precision

Precision, a vital metric in classification tasks, gauges the accuracy of positive predictions by measuring the proportion of correctly identified positive instances among those predicted as positive, excluding false positives. Its significance lies in scenarios where false positives carry substantial ramifications ([9], [10]). Nonetheless, a comprehensive evaluation of classification models necessitates the incorporation of complementary metrics like recall, accuracy, and F1 score especially in domains where false negatives hold critical implications, such as medical diagnosis or fraud detection. Precision is mathematically expressed as follows:

$$Precision = \frac{TP}{TP+FP} \tag{9}$$

3.1.4. Recall

Recall, a prevalent evaluation metric in classification tasks, quantifies the model's capability to accurately identify all positive cases by measuring the proportion of true positive predictions among all actual positive cases in the dataset, avoiding false negatives. Its significance is particularly pronounced in contexts where false negatives hold significant implications. However, to comprehensively assess the model's performance, considering precision alongside recall becomes imperative, especially in scenarios like spam email detection, where false positives bear substantial consequences. Recall is mathematically expressed as follows:

$$Recall = \frac{TP}{TP+FN} \tag{10}$$

3.1.5. F1 Score

The F1 score, a widely adopted metric in classification tasks, offers a balanced evaluation by harmonizing recall and precision into a single value. With equal consideration of both metrics, the F1 score, ranging from 0 to 1, indicates superior performance, particularly in scenarios where recall and precision hold equal importance. Nonetheless, being a general metric, it is advisable to complement the F1 score with other relevant metrics to gain a comprehensive perspective on the performance of the model, considering the specific requirements of the task. The following equation determines the F1 score:

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{11}$$

3.1.6. Area under the Receiver Operating Characteristic Curve (AUC ROC)

The Area under the Receiver Operating Characteristic Curve (AUC ROC) is a widely employed evaluation statistic in classification tasks, gauging the effectiveness of a binary classification model by plotting the True Positive Rate (TPR) versus the False Positive Rate (FPR) at various settings for the threshold. Ranging from 0 to 1, a higher AUC ROC value signifies superior performance, remaining advantageous over other metrics by its robustness to class disparity, comprehensive assessment across threshold settings, and ease of interpretation for non-technical stakeholders. Nevertheless, a potential limitation lies in its focus on prediction order rather than absolute values, which may lead to varying implications in specific applications even when classifiers possess the same AUC ROC value.

3.1.7. Matthews Correlation Coefficient

Matthews Correlation Coefficient (MCC) serves as a valuable indicator in binary classification evaluations, capturing the correlation between actual and predicted class labels while encompassing all possible outcomes. Ranging from -1 to +1, MCC offers advantages over other metrics by comprehensively considering positive and negative examples, proving robustness in handling imbalanced datasets, and accommodating class size variations. However, its interpretation may require additional effort, and as a correlation coefficient, it does not convey information regarding the magnitude or direction of classifier errors. Matthew's Correlation Coefficient is calculated as follows:

$$MCC = \frac{TN \times TP - FN \times FP}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{12}$$

3.2. Results

Table 1. Performance Evaluation of 60:40 Split

Classifier	Accuracy	Precision	Recall	F1 Score	AUC-ROC Score	MCC	TP	TN	FP	FN
Logistic Regression	0.9743	0.0567	0.9162	0.1067	0.9798	0.2244	175	110819	2913	16
Decision Tree	0.9977	0.4068	0.8115	0.5420	0.9048	0.5736	155	113506	226	36
Random Forest	0.9995	0.8526	0.8482	0.8504	0.9760	0.8501	162	113704	28	29
Gradient Boosting	0.9882	0.1144	0.8953	0.2028	0.9826	0.3177	171	112408	1324	20
XGBoost	0.9899	0.1312	0.8900	0.2286	0.9825	0.3395	170	112606	1126	21
SVM	0.9726	0.0531	0.9110	0.1003	0.9418	0.2162	174	110627	3105	17

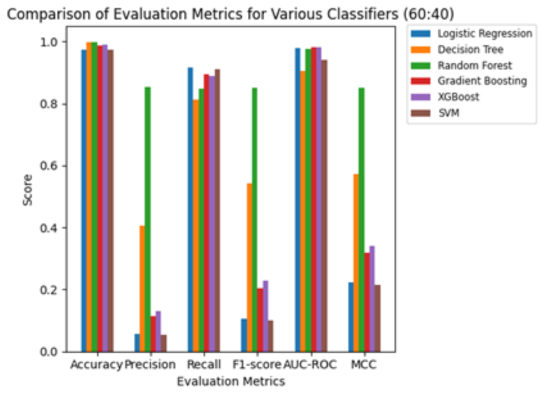


Fig. 4. Comparison of Evaluation Metrics for Various Classifiers (60:40)

The table (Table 1.) and the figure (Fig. 4.) exhibit the model's evaluation considering several evaluation measures for the dataset's 60:40 split. Some conclusions that can be drawn from them are:

1) With accuracy scores of 0.9995 and 0.9977, the Random Forest and Decision Tree models are the most accurate.
2) The Random Forest model has the highest precision, with a score of 0.8526.
3) The Logistic Regression model obtains the greatest recall value of 0.9162, and the SVM model follows close behind with a score of 0.9110.
4) The Random Forest model has the highest F1 score, with a score of 0.8504.
5) The Gradient Boosting and XGBoost models have the highest AUC ROC scores, with scores of 0.9826 and 0.9825, respectively.
6) The Random Forest model has the highest MCC score, with a score of 0.8501.

In summary, the Random Forest model performs best overall based on the provided data, with high scores across multiple performance metrics.

Table 2. Performance Evaluation of 70:30 Split

Classifier	Accuracy	Precision	Recall	F1 Score	AUC-ROC Score	MCC	TP	TN	FP	FN
Logistic Regression	0.9733	0.0529	0.9338	0.1002	0.9811	0.2189	127	83034	2273	9
Decision Tree	0.9971	0.3282	0.7794	0.4619	0.8884	0.5046	106	85090	217	30
Random Forest	0.9995	0.8357	0.8603	0.8478	0.9810	0.8477	117	85284	23	19
Gradient Boosting	0.9873	0.1046	0.9265	0.1880	0.9860	0.3091	126	84229	1078	10
XGBoost	0.9890	0.1182	0.9191	0.2096	0.9892	0.3275	125	84375	932	11
SVM	0.9741	0.0546	0.9338	0.1031	0.9540	0.2223	127	83106	2201	9

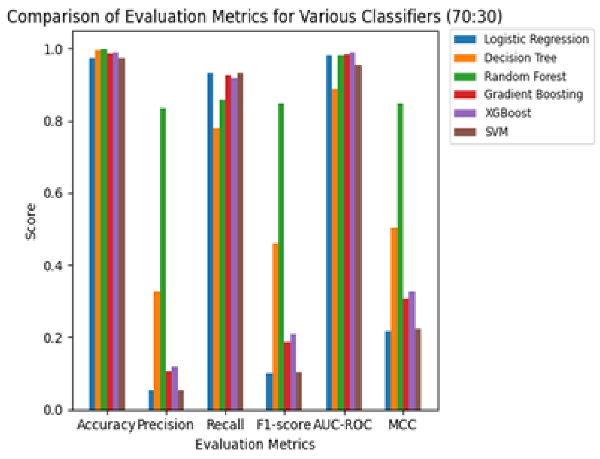


Fig. 5. Comparison of Evaluation Metrics for Various Classifiers (70:30)

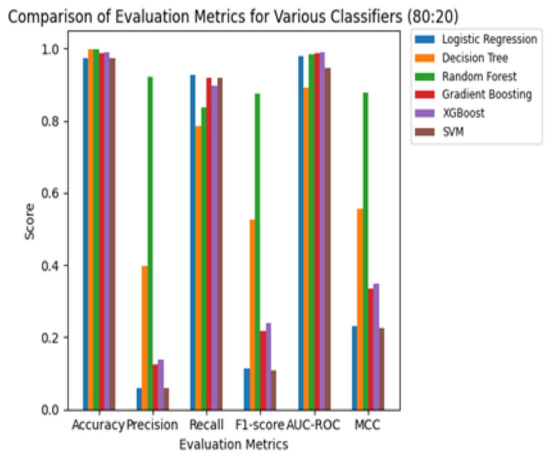


Fig. 6. Comparison of Evaluation Metrics for Various Classifiers (80:20)

The table (Table 2.) and the figure (Fig. 5.) exhibit the model's evaluation considering several evaluation measures for the dataset's 70:30 split. Some conclusions that can be drawn from them are:

1) With accuracy scores of 0.9995 and 0.9971, the Random Forest and Decision Tree models are the most accurate.
2) The Random Forest model has the highest precision, with a score of 0.8357.
3) The Logistic Regression and SVM models obtain the greatest recall value of 0.9338 each.
4) The Random Forest model has the highest F1 score, with a score of 0.8478.
5) The XGBoost and Gradient Boosting models have the highest AUC ROC scores, with scores of 0.9892 and 0.9860, respectively.
6) The Random Forest model has the highest MCC score, with a score of 0.8477.

Overall, the Random Forest model performs the best.

Table 3. Performance Evaluation of 80:20 Split

Classifier	Accuracy	Precision	Recall	F1 Score	AUC-ROC Score	MCC	TP	TN	FP	FN
Logistic Regression	0.9749	0.0600	0.9286	0.1128	0.9782	0.2326	91	55439	1425	7
Decision Tree	0.9976	0.3969	0.7857	0.5274	0.8918	0.5574	77	56747	117	21
Random Forest	0.9996	0.9213	0.8367	0.8770	0.9846	0.8778	82	56857	7	16
Gradient Boosting	0.9887	0.1240	0.9184	0.2184	0.9889	0.3352	90	56228	636	8
XGBoost	0.9902	0.1381	0.8980	0.2394	0.9892	0.3501	88	56315	549	10
SVM	0.9742	0.0580	0.9184	0.1090	0.9463	0.2272	90	55401	1463	8

The table (Table 3.) and the figure (Fig. 6.) exhibit the model's evaluation considering several evaluation measures for the dataset's 80:20 split. Some conclusions that can be drawn from them are:

1) With accuracy scores of 0.9996 and 0.9976, the Random Forest and Decision Tree models are the most accurate.
2) The Random Forest model has the highest precision, with a score of 0.9213.
3) The Logistic Regression model obtain the greatest recall value of 0.9286.
4) The Random Forest model has the highest F1 score, with a score of 0.8770.
5) The XGBoost and Gradient Boosting models have the highest AUC ROC scores, with scores of 0.9892 and 0.9889, respectively.
6) The Random Forest model has the highest MCC score, with a score of 0.8778.

Overall, the Random Forest models is the top performer based on the metrics.

4. Conclusion

In recent years, the application of machine learning algorithms for the detection of credit card fraud has proven to be quite successful. This study investigated and analyzed the performance of many common machine learning methods for detecting credit card fraud, with an emphasis on the influence of the SMOTE methodology on the performance of the model. SMOTE greatly enhanced the model's performance, with a noticeable reduction in false positives and false negatives. Among the various algorithms implemented, Random Forest performed the best, with a recall of 0.8482, precision of 0.8526 and F1 score of 0.8504 in the 60:40 split. It had a recall of 0.8603, precision of 0.8357 and F1 score of 0.8478 for the 70:30 split. For the

80:20 split, it had a recall of 0.8367, precision of 0.9213 and F1 score of 0.8770. However, it is worth noting that Logistic Regression, Gradient Boosting, XGBoost, and SVM also achieved good results in this context. In the future, there is plenty of room for more study in this field. For instance, the application of deep learning algorithms or hybrid models that combine multiple machine learning techniques can be explored to improve performance further. Furthermore, future work might concentrate on the creation of real-time detection and mitigation of credit card fraud systems capable of processing enormous volumes of data in real-time and preventing fraudulent transactions.

Author contributions

Rajani P.K.: Conceptualization, Methodology, Writing-Reviewing, Validation **Roshan Mathew:** Conceptualization, Methodology, Writing-Original draft preparation, Software, Validation **Atharva Walawalkar:** Conceptualization, Methodology, Writing-Original draft preparation, Software, Validation **Prathmesh Patil:** Data curation, Writing-Reviewing and Editing, Visualization, Validation,**Ujwal Shirode:** Writing-Reviewing, Validation **Ajjay Gaadhe:** Writing-Reviewing, Validation

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] S. Makki, Z. Assaghir, Y. Taher, R. Haque, M. -S. Hacid and H. Zeineddine, "An Experimental Study With Imbalanced Classification Approaches for Credit Card Fraud Detection," in *IEEE Access*, vol. 7, pp. 93010-93022, 2019, doi: 10.1109/ACCESS.2019.2927266.

[2] N. Malini and M. Pushpa, "Analysis on credit card fraud identification techniques based on KNN and outlier detection," *2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, Chennai, India, 2017, pp. 255-258, doi: 10.1109/AEEICB.2017.7972424.

[3] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic and A. Anderla, "Credit Card Fraud Detection - Machine Learning methods," *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, East Sarajevo, Bosnia and Herzegovina, 2019, pp. 1-5, doi: 10.1109/INFOTEH.2019.8717766.

[4] G. . Deshpande, R. . P. K., V. . Khandagle, and J. . Kolhe, "Comparison of Classification Algorithm for Crop Decision based on Environmental Factors using Machine Learning", *IJRITCC*, vol. 11, no. 9s, pp. 360–368, Aug. 2023.

[5] J. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," *2017 International Conference on Computing Networking and Informatics (ICCNI)*, Lagos, Nigeria, 2017, pp. 1-9, doi: 10.1109/ICCNI.2017.8123782.

[6] D. Tanouz, R. R. Subramanian, D. Eswar, G. V. P. Reddy, A. R. Kumar and C. V. N. M. Praneeth, "Credit Card Fraud Detection Using Machine Learning," *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2021, pp. 967-972, doi: 10.1109/ICICCS51141.2021.9432308.

[7] 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," in *IEEE Access*, vol. 10, pp. 39700-39715, 2022, doi: 10.1109/ACCESS.2022.3166891.

[8] P. Raghavan and N. E. Gayar, "Fraud Detection using Machine Learning and Deep Learning," *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, Dubai, United Arab Emirates, 2019, pp. 334-339, doi: 10.1109/ICCIKE47802.2019.90042

[9] R. . P. K., K. . Patil, B. . Marathe, P. . Mhaisane, and A. . Tundalwar, "Heart Disease Prediction using Different Machine Learning Algorithms ", *IJRITCC*, vol. 11, no. 9s, pp. 354–359, Aug. 2023.

[10] S. Dhankhad, E. Mohammed and B. Far, "Supervised Machine Learning Algorithms for Credit Card Fraudulent Transaction Detection: A Comparative Study," *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, Salt Lake City, UT, USA, 2018, pp. 122-125, doi: 10.1109/IRI.2018.00025.

[11] Z. Kazemi and H. Zarrabi, "Using deep networks for fraud detection in the credit card transactions," *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran, Iran, 2017, pp. 0630-0633, doi: 10.1109/KBEI.2017.8324876.

[12] P. . Patil, R. P. . K, K. . Salunkhe, H. . Patil, and R. . Kulkarni, "Covid-19 Detection For CT-scan Images Using Transfer Learning Models", *IJRITCC*, vol. 11, no. 8s, pp. 644–650, Aug. 2023.

[13] Ankit A. Mishra and C. Ghorpade, "Credit Card Fraud Detection on the Skewed Data Using Various Classification and Ensemble Techniques," *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal,

India, 2018, pp. 1-5, doi: 10.1109/SCEECS.2018.8546939.

[14] Rajani, P.K. and Khaparde, A. (2022). Video Error Concealment Using Particle Swarm Optimization. In Object Detection by Stereo Vision Images (eds R. Arokia Priya, A.V. Patil, M. Bhende, A. Thakare and S.
Wagh). https://doi.org/10.1002/9781119842286.ch4