Credit Card Fraud Detection Using ML Model

Ayagul Ormanova
Big Data Analysis
Department of Computing and Data Science
Astana, Kazakhstan
211291@astanait.edu.kz

Ayaulym Zhailmina
Big Data Analysis
Department of Computing and Data Science
Astana, Kazakhstan
211305@astanait.edu.kz

Aruzhan Zhumatay
Big Data Analysis
Department of Computing and Data Science
Astana, Kazakhstan
211240@astanait.edu.kz

Mansur Rakhymbay
Big Data Analysis
Department of Computing and Data Science
Astana, Kazakhstan
211570@astanait.edu.kz

***Abstract*—Credit card fraud is a significant concern for both financial institutions and individuals, as it often leads to substantial financial losses. To address this problem, machine learning (ML) models have been developed to detect fraudulent activities by analyzing transactional data. In this study, we employed logistic regression, decision tree, and random forest as ML models to identify fraudulent credit card transactions using a sample dataset that includes both normal and fraudulent transactions. To evaluate the effectiveness of these models, the dataset was split into training and testing data. Accuracy scores were calculated for both sets to measure the performance of each model. These findings provide a solid foundation for future research in credit card fraud detection using ML models. By implementing these models, financial institutions and individuals can better detect and prevent credit card fraud, ultimately reducing the associated financial losses.**

***Index Terms*—Logistic regression, Fraud detection, Credit card, Logistic regression, Decision tree, Random forest.**

I. Introduction

Credit card fraud detection using machine learning has become a critical task for the financial industry due to the rising number of fraudulent activities in recent years. Machine learning techniques are used to detect fraudulent transactions by identifying patterns and anomalies in the data [2]. The use of machine learning in fraud detection has been studied in several research papers, and various models have been proposed.

Credit card fraud is a broad term that encompasses the use of credit cards to purchase goods or services without the intention of paying for them. This type of fraud includes identity theft, identity fraud, and fraudulent activity. Fraud detection involves monitoring cardholder transaction behavior to identify any unauthorized activity. Traditional methods of fraud detection have been used for a long time but are often ineffective and time-consuming. The combination of machine learning and artificial intelligence is necessary to efficiently detect fraudulent activity [3].

The topic of credit card fraud detection using machine learning involves collecting data from numerous credit card users and then preprocessing and profiling the data [2]. By analyzing the data set, associations can be found using various machine learning algorithms, and inconsistencies such as fraudulent transactions can be identified. The fundamental principle behind machine learning-based fraud detection is training a model that can recognize legitimate transaction patterns and identify any deviations from those patterns.

For credit card fraud detection using machine learning, we obtained a dataset from Kaggle website named 'Credit Card Fraud Detection' [7]. The dataset consists of 31 columns, including the target column 'class' that needs to be predicted. The dataset comprises 416,679 data points or rows. It has two classes to be predicted: '0' or '1', representing 'normal' or 'fraudulent' transactions.

Credit card fraud occurs when someone uses another person's credit card without the knowledge of the card owner or issuer. This can happen through the theft of the physical card or the unauthorized access to important account information, such as the card account number. Such information is typically required by merchants during a legitimate transaction. Card numbers, usually called Primary Account Numbers (PAN), are often printed on the card. The back of the card has a magnetic stripe that holds the data in a format that machines can read. The stripe contains the following information:

1) Name of card holder
2) Card number
3) Expiration date
4) Verification/CVV code
5) Type of Card

There are many ways to commit credit card fraud. Fraudsters are skilled and fast individuals. One common method is Application Fraud, where a person provides false information to obtain a credit card. Another method is the unauthorized use of Lost and Stolen Cards, which accounts for a significant portion of credit card fraud. More sophisticated fraudsters create Fake and Altered Cards or use Skimming to copy information from one card to another. Another popular method involves

creating fake websites or cloning legitimate ones to trick people into revealing their credit card details unknowingly.

Credit card fraud detection is a complex problem due to the ever-evolving nature and patterns of fraudulent transactions [4]. Moreover, current machine learning models for detecting credit card fraud face difficulties in achieving high detection accuracy and addressing the imbalanced nature of credit card fraud datasets. As a result, it is imperative to develop machine learning models that can perform optimally and accurately detect credit card fraud.

The remainder of this paper is organized as follows:

Section 2: Methodology and Proposed Techniques. In this section, we delve into the main part of our study, presenting the methodology employed and the practical aspects of our approach. We outline the techniques used for credit card fraud detection, including logistic regression, decision tree, and random forest. The proposed techniques are described in detail, highlighting their relevance and potential advantages in tackling fraud detection challenges.

Section 3: Results and Discussions. This section focuses on presenting the results obtained from our experiments and their subsequent analysis. Performance metrics such as sensitivity, specificity, accuracy, and error rate are used to evaluate the effectiveness of the proposed techniques. We provide an interpretation of the results, discussing the implications and significance of our findings. Additionally, we explore potential future directions for further research and improvement in credit card fraud detection based on the outcomes of our study.

Section 4: Conclusion. In the final section, we draw our conclusions based on the insights gained from our research. We summarize the key findings, highlighting the performance of logistic regression, decision tree, and random forest in detecting credit card fraud. We emphasize the superiority of the random forest technique over logistic regression and decision tree. We discuss the implications of our study for the field of credit card fraud detection and suggest possible avenues for future investigations.

II. Main Part

A. Methodology

The methodological approach used in this study is a quantitative method. Observational data will be collected from a sample of credit card transactions from a financial institution. The purpose of collecting observational data from a sample of credit card transactions from a financial institution is to gain insights into the patterns and characteristics of normal and fraudulent transactions. The observational data will provide a snapshot of the actual behavior of credit card users, which can be used to develop machine learning models that can detect fraudulent activities accurately. By analyzing the data, we can identify patterns and anomalies in the transactions and use them to develop effective fraud detection models. This approach will help financial institutions to save themselves and their customers from financial losses caused by fraudulent transactions. The quantitative method will enable us to analyze the data and draw statistically significant conclusions, which can be used to make informed decisions about credit card fraud detection. The steps taken in this methodology are as follows:

Data collection. The dataset used in this study is a publicly available credit card transaction dataset from Kaggle [8].

Data preprocessing. The CSV file is read and the first 5 rows of the DataFrame are printed. The data is then handled for missing values and the statistical measures of the data are computed.

Data analysis. During the data analysis phase, various techniques are employed to evaluate the performance of the models. Sensitivity, specificity, accuracy, and error rate are the key variables used for this evaluation. Sensitivity measures the ability of the model to correctly identify fraudulent transactions, while specificity measures its ability to correctly identify normal transactions. Accuracy reflects the overall correctness of the model's predictions, while error rate quantifies the proportion of incorrect predictions made by the model.

Model development. In this study, we employed the Logistic Regression model as well as decision tree and random forest models for credit card fraud detection [6]. To evaluate the performance of each model, accuracy scores were calculated for both the training and testing datasets. To begin the analysis, the credit card transaction dataset was utilized, containing information on both normal and fraudulent transactions. The initial step involved exploring the dataset and gaining an understanding of its distribution. Subsequently, the dataset was divided into training and testing datasets, employing a stratified approach to maintain the proportional representation of normal and fraudulent transactions in both sets. For each model, the training data was utilized to train the respective model. Following the training process, the accuracy score was computed for both the training and testing datasets. This allowed us to assess how well each model performed in accurately classifying transactions as normal or fraudulent. By considering the accuracy scores for each model on both the training and testing datasets, we gained insights into the performance of logistic regression, decision tree, and random forest models in credit card fraud detection.

The methodology employed in this analysis follows a clear and structured approach to examining the credit card transaction dataset. It leverages quantitative methods and observations to gain insights. The obtained results indicate that the decision tree, random forest and logistic regression models show promising performance in predicting transaction types based on the dataset's features. These models can provide a more in-depth understanding of the dataset and potentially enhance the accuracy of predictions. By incorporating these models into the analysis, we can further refine our understanding of credit card fraud detection and develop more robust and precise models.

B. Practical Part

1) Programming language and Essential Libraries: For Credit card fraud detection using ML model, the programming language used is Python, which is widely recognized for its extensive support in the field of data science and machine learning. The implementation leverages essential libraries such

as NumPy [8] and Pandas [9] to facilitate efficient data manipulation, preprocessing, and analysis. NumPy, imported as np, provides a robust foundation for working with large, multi-dimensional arrays and offers a variety of mathematical operations to enable efficient numerical computations. On the other hand, Pandas, imported as pd, offers versatile data structures, including the DataFrame, which simplifies tasks related to structured data management, such as loading datasets and performing data preprocessing and exploration. Scikit-learn [10], imported as sklearn, is employed as a prominent machine learning library in Python. It encompasses a diverse range of algorithms and tools for various tasks, including data preprocessing, model training, and evaluation. Built upon other scientific libraries like NumPy and SciPy, Scikit-learn provides a comprehensive environment for developing and deploying machine learning models.

Within the Scikit-learn library, specific modules and functions are utilized for Credit card fraud detection, such as:

Train test split [10]: This function from the model selection module is used to split the dataset into separate training and testing subsets. It ensures that the model is trained on a portion of the data and evaluated on unseen data, enabling an assessment of its generalization performance.

Logistic Regression [11]: This class from the linear model module is employed to fit a logistic regression model to the given dataset. Logistic regression is a popular classification algorithm that predicts the probability of a binary outcome, making it suitable for identifying fraudulent and non-fraudulent transactions.

Decision Tree: In this analysis, we utilize the decision tree algorithm. Decision trees are widely used for classification tasks and work by partitioning the dataset based on different attributes to create a tree-like structure. Each internal node represents a test on a specific attribute, while each leaf node represents a class label. This algorithm is well-suited for credit card fraud detection due to its ability to capture complex relationships within the dataset.

Random Forest: Another algorithm explored in this analysis is the random forest. Random forest combines multiple decision trees to form an ensemble model. Each tree is built on a randomly selected subset of the dataset, and the final prediction is made by aggregating the predictions of individual trees. This technique helps improve the accuracy and robustness of the model by reducing overfitting and capturing different aspects of the data.

Accuracy score [12]: This function from the metrics module is used to calculate the accuracy of the classification model. It compares the predicted labels of the model with the true labels and determines the percentage of correct predictions. A high accuracy score on both the training and testing datasets indicates that the model is performing well and can be useful for credit card fraud detection.

C. Proposed Technique

This paper employs various techniques to detect fraud in the credit card system. A comparison is made among different machine learning algorithms, including Logistic Regression, Decision Trees, and Random Forest, to identify the most suitable algorithm that can be adopted by credit card merchants for fraud detection. The architectural diagram, depicted in Figure 1, illustrates the overall system framework.

In order to identify the most suitable algorithm for the given dataset, we outline the processing steps employed in this analysis. By following these steps, we aim to determine the algorithm that yields the most effective results in terms of accuracy and performance:

Step 1: Read the dataset.

Step 2: Random Sampling is done on the data set to make it balanced.

Step 3: Divide the dataset into two parts i.e., Train dataset and Test dataset.

Step 4: Feature selection are applied for the proposed models.

Step 5: Accuracy and performance metrics has been calculated to know the efficiency for different algorithms.

Step 6: Then retrieve the best algorithm based on efficiency for the given dataset.

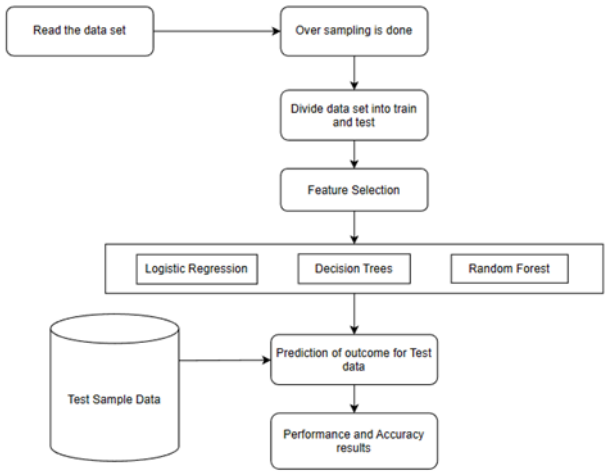

Fig. 1. System Architecture

1) Logistic Regression: Logistic Regression is a classification algorithm commonly used to predict binary values (e.g., 1/0, Yes/No, True/False) based on a set of independent variables. In order to represent binary or categorical values, dummy variables are employed. In the case of logistic regression, which is a special case of linear regression, the dependent variable is transformed using the logarithm of odds when it is categorical. This allows for the prediction of the probability of an event occurring by fitting the data to a logistic function. Such us:

$$O = \frac{e^{I_0 + I_1 x}}{1 + e^{I_0 + I_1 x}} \tag{1}$$

In this equation:

O represents the output or outcome.

e represents the mathematical constant Euler's number ($e \approx 2.71828$).

I_0 and I_1 are parameters or variables.

x is an input variable.

The equation calculates the value of O using the exponentiation of e to the power of $I_0 + I_1 x$. This value is divided by 1 plus e raised to the power of $I_0 + I_1 x$.

$$y = \frac{e^{b_0 + b_1 x}}{1 + e^{b_0 + b_1 x}} \tag{2}$$

Logistic regression begins with a simplified form of the linear regression equation, where the dependent variable is transformed using a link function. To initiate logistic regression, we first express the simple linear regression equation with the dependent variable enclosed within a link function.

$$A(O) = \beta_0 + \beta(x) \tag{3}$$

where,

1) $A()$ is the link function;
2) O is the outcome variable;
3) x is the dependent variable;

A function is created by combining two elements:

1) Probability of Success (pr);
2) Probability of Failure (1-pr);

The probability, pr, must meet the following criteria:

1) The probability must always be positive, since $p \geq 0$.
2) The probability must always be less than or equal to 1, since $pr \leq 1$.

By applying the exponential function, the first criterion ensures that the value is always greater than or equal to 1.

$$pr = \exp(\beta_0 + \beta(x)) = e^{\beta_0 + \beta(x)} \tag{4}$$

To meet the second criterion, the exponential value is divided by adding 1 to it. This division ensures that the resulting value is less than or equal to 1.

$$pr = \frac{e^{\beta_0 + \beta(x)}}{e^{\beta_0 + \beta(x)} + 1} \tag{5}$$

Logistic regression employs a logistic function to model the relationship between the predictors and the response variable. In this regression, a cost function is used to measure the error by comparing the model's predicted response with the true value. The cost function helps quantify the extent to which the logistic regression model deviates from accurately representing the observed data. By minimizing this error through the optimization process, the logistic regression model strives to provide the best possible fit to the data and make accurate predictions.

$$X(\theta) = -\frac{1}{m}\left(\sum_i y_i \log(h_\theta(x_i)) + (1 - y_i)\log(1 - h_\theta(x_i))\right) \tag{6}$$

where,

- $h_\theta(x_i)$ represents the logistic function;
- y_i is the outcome variable and Gradient descent is a learning algorithm;

2) Decision Tree Algorithm: A decision tree is a supervised learning algorithm commonly used for solving classification problems. It is applicable to both categorical and continuous input and output variables. This technique involves dividing the population or sample into multiple homogeneous sets, also known as sub-populations. The division is based on the most significant splitter or differentiator found within the input variables. By using this method, we can effectively organize and classify data based on key factors, enabling accurate predictions and analysis.

There are two types of decision trees:

1) Categorical Variable Decision Tree: This type of decision tree is used when the target variable (the variable we are trying to predict) is categorical. It is designed specifically to handle situations where the target variable falls into discrete categories.
2) Continuous Variable Decision Tree: This type of decision tree is used when the target variable is continuous, meaning it takes on a range of numerical values. It is designed to handle situations where the target variable is measured on a continuous scale.

Decision Tree Terminology:

1) Root Node: The root node represents the entire population or sample, which is then divided into two or more homogeneous subsets.
2) Splitting: Splitting refers to the process of dividing a node into two or more sub-nodes based on certain criteria or conditions.
3) Decision Node: When a sub-node further splits into additional sub-nodes, it is referred to as a decision node.
4) Leaf/ Terminal Node: A leaf or terminal node is a node that does not split any further. It represents the final outcome or classification.
5) Pruning: Pruning is the process of removing sub-nodes from a decision node. It is the opposite of splitting and helps simplify the decision tree.
6) Branch / Sub-Tree: A branch or sub-tree refers to a specific section of the entire decision tree.
7) Parent and Child Node: A parent node is a node that is divided into sub-nodes, while the sub-nodes are referred to as the children of the parent node.

Let's denote the features of a card as $X = \{x_1, x_2, \ldots, x_n\}$, where x_i represents a specific feature. The target variable, indicating the card type, is denoted as Y.

The decision tree model learns to make decisions based on the values of the features. At each internal node of the tree, a splitting criterion is applied to determine which branch to follow. This criterion is typically based on some measure of purity or information gain.

The process continues recursively until reaching a leaf node, which represents a predicted card type. The decision tree

partitions the feature space into distinct regions, with each region corresponding to a specific card type.

The decision tree can be represented as a set of rules. Each internal node represents a decision based on a specific feature, and each leaf node represents a predicted card type.

Mathematically, the decision tree model can be represented as a function $f : X \rightarrow Y$, where $f(X)$ predicts the card type based on the given features.

Decision trees utilize various algorithms to determine whether to split a node into multiple sub-nodes. This process of creating sub-nodes aims to enhance the homogeneity within each resulting sub-node. Essentially, it seeks to increase the purity of the node in relation to the target variable being analyzed. The decision tree algorithm evaluates all available variables to determine the optimal split that yields the most homogeneous sub-nodes.

To achieve this, several metrics can be employed, including:

1) Gini Index: This metric quantifies the impurity of a node by measuring the probability of misclassifying a randomly selected element from the node. Lower Gini Index values indicate a higher level of purity in the resulting sub-nodes.
2) Information Gain: Information Gain measures the reduction in entropy (i.e., the degree of disorder) achieved by splitting a node on a specific variable. It evaluates how much information is gained from the split, with higher values indicating more informative splits.
3) Chi-Square: This statistical test is used to assess the independence of variables. In decision trees, the Chi-Square metric determines the significance of the relationship between a potential split and the target variable. A lower p-value suggests a stronger association between the split and the target variable.
4) Reduction of Variance: This metric, also known as the variance impurity or mean squared error, measures the reduction in variance achieved by splitting a node. It aims to minimize the overall variance within each resulting sub-node, resulting in more homogeneous sub-nodes.

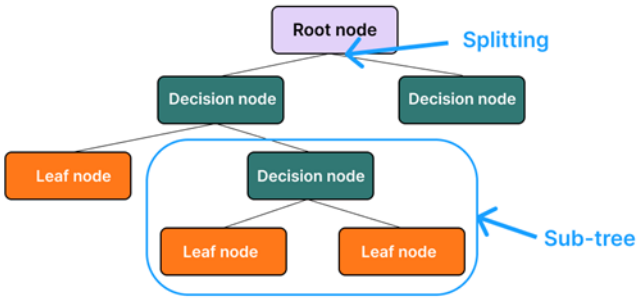

Fig. 2. Decision tree Model

3) Random Forest: Random Forest (RF) is an ensemble technique that belongs to the group learning category, applicable for both classification and regression tasks. This method utilizes deep trees to effectively capture irregular patterns in the data. When deep trees learn the same part of the training sample, RF calculates the average of the variation in their values. This averaging process helps reduce the potential variance in the predictions.

The training data consists of a set of predictors $(p_1, p_2, \ldots, p_n)$ and corresponding responses $(q_1, q_2, \ldots, q_n)$. To build a Random Forest model, the algorithm performs bagging (sampling with replacement) multiple times. In each iteration $(x = 1, 2, \ldots, X)$, a random sample is chosen from the training data, and the trees are built based on these samples.

For $x = 1, \ldots, X$:

$$\frac{1}{X} \sum_{x=1}^{X} f(x)(R) \tag{7}$$

Working of random forest:

The Bagging Algorithm is utilized to create random samples from a given dataset, D1, which consists of n rows and m columns. A new dataset, D2, is formed by randomly sampling n cases from the original data with replacement. Within D1, approximately one-third of the rows are left out, forming what is known as Out of Bag (OOB) samples. D2 is then trained using these models, while the OOB samples are used to estimate the error in an unbiased manner. From the original m columns, a smaller subset M ¡¡ m is randomly selected at each node in the dataset. The default choice for M varies depending on the type of tree: m/3 for regression trees and sqrt(m) for classification trees. Unlike a regular decision tree, no pruning takes place in a random forest model. Each tree in the random forest is grown to its full extent. Pruning is a technique used in decision trees to prevent overfitting, where a subtree is selected to minimize the test error rate. Cross-validation is employed to determine the test error rate of the subtree. In a random forest, multiple trees are grown, and the final prediction is obtained by averaging or voting among the predictions of these trees.

To find the best algorithm for credit card fraud detection, the following steps can be followed:

1) Import the dataset containing credit card transaction data.
2) Convert the dataset into the data frame format for further analysis.
3) Perform random oversampling using the ROSE package to address class imbalance issues.
4) Determine the proportion of data to be allocated for training and testing.
5) Allocate 70% of the data for training and the remaining portion for testing.
6) Assign the training dataset to the machine learning models.
7) Choose one algorithm from three different options and create the model.
8) Make predictions for the test dataset using each algorithm.

9) Calculate the accuracy of each algorithm's predictions.
10) Apply a confusion matrix to evaluate the performance of each algorithm for each variable.
11) Compare the algorithms across all variables and identify the best-performing algorithm.

By following these steps, one can systematically compare and identify the best algorithm for credit card fraud detection based on its performance and suitability for the given dataset.

4) Evaluating Model Performance: The accuracy score is a useful metric to assess the performance of a model in classifying transactions as either fraudulent or non-fraudulent [10]. A higher accuracy score indicates that the model is effective at correctly classifying most transactions. However, it is important to note that accuracy alone may not provide a comprehensive evaluation, particularly when working with imbalanced datasets where the number of fraudulent transactions is significantly smaller than the number of non-fraudulent transactions. In such cases, other evaluation metrics like precision, recall, and F1 score are necessary to gain a deeper understanding of the model's performance in correctly identifying fraudulent transactions. These additional metrics take into account false positives and false negatives, which can be particularly important in fraud detection scenarios.

5) Model Evaluation: The accuracy score provides a measure of how well the model is able to classify transactions as fraudulent or non-fraudulent.

6) Implementation of Main Pseudocode Algorithms: In this section, we delve into the practical aspect of applying the main pseudocode algorithms. We provide detailed explanations and examples of how these algorithms are implemented to solve specific problems or tasks.

By providing concrete examples and code snippets, we aim to facilitate a clear understanding of the main pseudocode algorithms and their implementation in real-world scenarios. We also discuss any considerations or optimizations that may be relevant to ensure efficient and accurate results.

Algorithm 1 Creating a Logistic Regression Model

function CREATE_LOGISTIC_REGRESSION_MODEL(max_iterations)
  Parameters:
  $max\_iterations$: Maximum number of iterations for convergence
  $model \leftarrow$ LogisticRegression($max\_iter = max\_iterations$)
  $model$.fit(X_train, Y_train)

  Return $model$
end function

III. Results and Discussions

In this section, we present the results of the credit card fraud detection using the logistic regression model, as well as the decision tree and random forest models, and discuss their implications. The performance of each model is evaluated based on various metrics and compared to the baseline and other existing methods to determine their effectiveness in detecting fraudulent transactions. By considering the results and comparing the performance of these models, we can gain insights into the strengths and limitations of each approach, enabling us to make informed decisions about the most suitable method for credit card fraud detection.

Algorithm 2 Train-Test Split

function TRAIN_TEST_SPLIT(X, Y, $test\_size$, $stratify$, $random\_state$)
  Inputs:
  X: Input features
  Y: Target variable
  $test\_size$: Proportion of data to be used for testing
  $stratify$: Ensures class balance in the split
  $random\_state$: Seed for random number generation
  Calculate the number of instances for testing
  $test\_instances \leftarrow test\_size \times$ number_of_instances_in_data(X)
  Set the random seed for reproducibility
  set_random_seed($random\_state$)
  Shuffle the data randomly
  $shuffled\_X, shuffled\_Y \leftarrow$ random_shuffle(X, Y)
  Select the first $test\_instances$ **instances for testing**
  $X\_test \leftarrow$ get_first_n_instances($shuffled\_X, \leftarrow$ test_instances)
  $Y\_test \leftarrow$ get_first_n_instances($shuffled\_Y, \leftarrow$ test_instances)
  Select the remaining instances for training
  $X\_train \leftarrow$ get_remaining_instances($shuffled\_X, \leftarrow test\_instances$)
  $Y\_train \leftarrow$ get_remaining_instances($shuffled\_Y, \leftarrow test\_instances$)
  Return the training and testing sets
  return $X\_train, X\_test, Y\_train, Y\_test$
end function

A. Performance metrics

The performance of a binary classifier can be assessed using basic performance measures derived from the confusion matrix. The confusion matrix is a 2x2 table that presents the outcomes produced by the classifier. From this matrix, several important measures are computed, including sensitivity, specificity, accuracy, and error rate. These measures provide valuable insights into the performance of the classifier and help evaluate its effectiveness in correctly classifying instances. By analyzing the confusion matrix and deriving these performance measures, we can gain a better understanding of the classifier's capabilities and make informed decisions about its utility in practical applications.

1) Accuracy: Accuracy is a measure of how well a model predicts the correct outcomes in a dataset. It is calculated by dividing the total number of correct predictions (both true

Algorithm 3 Decision Tree Classification

function DECISION_TREE_CLASSIFICATION(X_{train}, X_{test}, Y_{train}, Y_{test})
  Initialize object for DecisionTreeClassifier class
  $dt\_classifier \leftarrow$ DecisionTreeClassifier($criterion = 'gini'$)
  Print "Model training starts....."
  $dt\_classifier$.fit($X_{\text{train}}, Y_{\text{train}}$.values.ravel())
  Print "Model training completed"
  $acc\_score \leftarrow dt\_classifier$.score($X_{\text{test}}, Y_{\text{test}}$)
  Print "Accuracy of model on test dataset: ", acc_score
  $y\_pred \leftarrow dt\_classifier$.predict($X_{\text{test}}$)
  $cm \leftarrow$ confusion_matrix(Y_{test}, y_pred)
  Print "Confusion Matrix:"
  Print cm
  Print "Classification Report:"
  Print classification_report(Y_{test}, y_pred)
  $aroc\_score \leftarrow$ roc_auc_score(Y_{test}, y_pred)
  Print "AROC score: ", $aroc\_score$
  Visualize Confusion Matrix
  Plot cm using heatmap with annotations and colormap 'Blues'
  Set title as "Confusion Matrix"
  Set x-label as "Predicted labels"
  Set y-label as "True labels"
  Show the plot
end function

Algorithm 4 Random Forest Classifier

function RANDOM_FOREST_CLASSIFIER(X_{train}, X_{test}, Y_{train}, Y_{test})
  Initialize object for RandomForestClassifier class
  $rf\_classifier \leftarrow (n\_estimators = 50)$
  Print "Model training starts....."
  $rf\_classifier$.fit($X_{\text{train}}, Y_{\text{train}}$.values.ravel())
  $acc\_score \leftarrow rf\_classifier$.score($X_{\text{test}}, Y_{\text{test}}$)
  Print "Accuracy of model on test dataset: ", acc_score
  $y\_pred \leftarrow rf\_classifier$.predict($X_{\text{test}}$)
  $cm \leftarrow$ confusion_matrix(Y_{test}, y_pred)
  Print "Confusion Matrix :-"
  Print cm
  Print "Classification Report :-"
  Print classification_report(Y_{test}, y_pred)
end function

positives and true negatives) by the total number of instances in the dataset. Accuracy can also be calculated as 1 minus the error rate.

The formula for accuracy is:

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

where,

- TP refers to true positives, which are the instances correctly identified as positive.

- TN refers to true negatives, which are the instances correctly identified as negative.

- FP refers to false positives, which are the instances incorrectly classified as positive.

- FN refers to false negatives, which are the instances incorrectly classified as negative.

By summing the true positives and true negatives and dividing it by the total number of instances, which includes true positives, false positives, true negatives, and false negatives, we can calculate the accuracy of a model's predictions.

2) Precision: Precision is a performance assessment metric that quantifies the ratio of correctly identified positive instances (true positives) to the total number of instances identified as positive, including both true positives and false positives. It can be calculated using the following formula:

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

In this formula:

- TP represents true positives, which are the instances correctly identified as positive.

- FP represents false positives, which are the instances incorrectly classified as positive.

By calculating precision, we can determine the accuracy of positive predictions made by a model and assess its ability to avoid false positives. A higher precision score indicates a lower rate of false positives and a higher level of confidence in the model's positive predictions.

3) F1 score: The f-measure, also known as the F1 score, is a metric that combines both precision and recall to provide a balanced evaluation of a model's performance. It can be calculated as the weighted average of precision and recall using the following formula:

$$F = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

In this formula:

- Precision represents the ratio of true positives to the sum of true positives and false positives.

- Recall represents the ratio of true positives to the sum of true positives and false negatives.

The f-measure takes into account both precision and recall, ensuring that the evaluation considers both the accuracy of positive predictions (precision) and the ability to identify all positive instances (recall). By calculating the f-measure, we can obtain a single metric that provides a comprehensive assessment of a model's performance in handling both false positives and false negatives.

4) Recall: Recall, also known as sensitivity, is a metric that measures the proportion of true positive instances correctly identified by a model. It is calculated as the ratio of true positives to the sum of true positives and false negatives. The formula for recall is as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

In this formula:

- TP represents true positives, which are the instances correctly identified as positive.

- FN represents false negatives, which are the instances incorrectly classified as negative.

By calculating recall, we can assess the model's ability to correctly identify positive instances and minimize false negatives. A higher recall score indicates a higher rate of correctly identifying positive instances and a lower rate of false negatives, suggesting a model's effectiveness in capturing all relevant positive instances.

B. Top 3 algorithms for Fraud Detection

The study focuses on utilizing the top three machine learning algorithms for credit card fraud detection. These algorithms include:

1. Logistic Regression
2. Decision Tree
3. Random Forest

These algorithms are widely used in various areas of machine learning research and development. In addition to credit card fraud detection, they can be applied to association analysis, clustering, classification, statistical learning, and link mining.

These topics represent critical areas of interest in machine learning, with extensive research and development efforts dedicated to improving their performance and applicability. By incorporating these algorithms into the study, the researchers aim to contribute to the advancement of credit card fraud detection and address the challenges associated with it.

C. The accuracy of algorithms

In this study, the performance of three machine learning algorithms for detecting credit card fraud is investigated. The evaluation of these algorithms is based on metrics such as accuracy, error rate, sensitivity, and specificity. To assess the algorithms, 70% of the dataset is utilized for training, while the remaining 30% is used for testing and validation.

Table 1 presents the results obtained for different variables using the three algorithms. The accuracy achieved by logistic regression is 98.1%, while the decision tree and random forest classifiers achieve accuracies of 93.8% and 99.01% respectively. Comparatively, the random forest algorithm outperforms both logistic regression and decision tree techniques.

TABLE I
FEATURE SELECTION RESULTS

2*Variable	Model Accuracy (%)		
	Logistic Regression	Decision Tree	Random Forest
All Variables	98.1	93.8	99.01

D. Implications and Future Directions

These results have important implications for the financial industry, as they demonstrate the potential of machine learning models, specifically logistic regression, decision tree, and random forest, in mitigating credit card fraud. By accurately identifying fraudulent transactions, financial institutions can take proactive measures to prevent financial losses and protect their customers' assets. Moreover, these models can be integrated into real-time transaction monitoring systems, enabling timely detection and prevention of fraudulent activities.

However, there are several areas for future improvement and exploration. Firstly, the imbalance in the dataset, with a smaller number of fraudulent transactions, poses a challenge for accurate fraud detection. Addressing this issue through techniques such as oversampling, undersampling, or using ensemble methods like random forests or advanced anomaly detection algorithms could potentially enhance the models' performance.

The current study focused on logistic regression, decision tree, and random forest as individual models for credit card fraud detection. Investigating the performance of other machine learning algorithms, such as support vector machines or deep learning models, could provide valuable insights into their suitability for this task. Comparative studies involving multiple models can help identify the most effective approach for credit card fraud detection.

It is essential to continuously update and adapt the credit card fraud detection models to stay ahead of evolving fraud techniques. Incorporating data from new fraud patterns, regularly retraining the models, and conducting frequent evaluations are crucial steps to ensure the models' effectiveness in real-world scenarios.

CONCLUSION

This paper focuses on the detection of fraud in credit card systems using machine learning techniques, specifically logistic regression, decision tree, and random forest. The performance of these methods is evaluated using metrics such as sensitivity, specificity, accuracy, and error rate.

The accuracy results obtained for logistic regression, decision tree, and random forest classifiers are 98.1%, 93.8%, and 99.01% respectively. Through comparative analysis, it is determined that the random forest classifier outperforms both logistic regression and decision tree in terms of accuracy.

By employing these machine learning techniques, the proposed system demonstrates its ability to effectively identify fraudulent transactions in the credit card system. The findings suggest that the random forest classifier is the most suitable method for fraud detection among the evaluated models.

REFERENCES

[1] E. Ileberi, Y. Sun, and Z. Wang, "A machine learning based credit card fraud detection using the ga algorithm for feature selection," Journal of Big Data, vol. 9, no. 1, pp. 1–17, 2022. [Online]. Available: http://dx.doi.org/10.1186/s40537-022-00573-8

[2] P. Sharma, S. Banerjee, D. Tiwari, and J. Patni, "Machine learning model for credit card fraud detection-a comparative analysis," Int. Arab J. Inf. Technol., vol. 18, no. 6, pp. 789–796, 2021. [Online]. Available: https://iajit.org/portal/images/year2021/no6/19792.pdf

[3] A. Thennakoon, C. Bhagyani, S. Premadasa, S. Mihiranga, and N. Kuruwitaarachchi, "Real-time credit card fraud detection using machine learning," in 2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence), ser. IPTPS '01. IEEE, 2019, pp. 488–493. [Online]. Available: http://dx.doi.org/10.1109/CONFLUENCE

[4] S. Thudumu, P. Branch, J. Jin et al., “A comprehensive survey of anomaly detection techniques for high dimensional big data,” J Big Data, vol. 7, no. 1, p. 42, 2020. [Online]. Available: http://dx.doi.org/10.1186/s40537-020-00320-x

[5] S. Oprea, A. Bara, F. Puican, and I. Radu, “Anomaly detection with machine learning algorithms and big data in electricity consumption,” Sustainability, vol. 13, no. 19, 2021. [Online]. Available: http://dx.doi.org/10.3390/su131910963

[6] Logistic Regression, https://en.wikipedia.org/wiki/Logistic_regression

[7] Credit Card Fraud Detection: Anonymized credit card transactions labeled as fraudulent or genuine, https://www.kaggle.com/mlgulb/creditcardfraud

[8] NumPy. (n.d.). Retrieved from https://numpy.org/

[9] Pandas. (n.d.). Retrieved from https://pandas.pydata.org/

[10] Scikit-learn: Machine Learning in Python. (n.d.). Retrieved from https://scikit-learn.org/stable/index.html

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

[12] Scikit-learn: Accuracy Score. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

[13] McKinney, W., & Others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51-56).

[14] pandas. (n.d.). pandas.read_csv - pandas 1.3.1 documentation. Retrieved from https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

[15] K. Z. Ghafoor, S. Lee, and Y. Kim, ”A machine learning approach to credit card fraud detection,” Journal of Intelligent & Fuzzy Systems, vol. 39, no. 3, pp. 3169-3180, 2020.

[16] Pandas. (n.d.). DataFrame.describe. Retrieved from https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.describe.html

[17] Wes McKinney, ”Data Structures for Statistical Computing in Python,” Proceedings of the 9th Python in Science Conference, pp. 51-56, 2010.

[18] A. Anwar and S. Shabbir, ”Credit Card Fraud Detection Using Machine Learning: A Systematic Review,” in IEEE Access, vol. 9, pp. 36136-36154, 2021.

[19] Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

[20] Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825-2830.

[21] Scikit-learn developers. (n.d.). accuracy_score. scikit-learn: machine learning in Python. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html