Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Detecting Financial Fraud Using Anomaly Detection Techniques: A Comparative Study of Machine Learning Algorithms

MD RASHED MOHAIMIN[1] ✉ Md Sumsuzoha[2], Md Amran Hossen Pabel[3] and Farhan Nasrullah[4]

[1]*Business Analytics, Gannon University*
[2]*Master of Science in Business Analytics, Trine University*
[3]*Department of Marketing, Wright State University*
[4]*Business Analytics, Gannon University*

Corresponding Author: MD RASHED MOHAIMIN, **E-mail**: mohaimin001@gannon.edu

| ABSTRACT

Financial fraud presents a substantial challenge to the American economy, culminating in substantial monetary losses and breaching the integrity of financial systems. The focal objective of this research paper was to resolve the prevalent issue of financial fraud detection in the USA by performing a comparative study of multiple machine learning algorithms, particularly concentrating on their anomaly detection capabilities. Experimentation was performed using various machine learning classifiers, such as logistic regression, random forest, Multi-layer perceptron, SVM, Naive Bayes, AdaBoost, decision tree, and KNN. Data utilized for this study was retrieved from Kaggle's website (https://www.kaggle.com/mlg-ulb/creditcardfraud). The five-metrics used for the performance evaluation were accuracy, precision, recall, F1-score, and confusion matrix. Decision Tree had superior performance at classification accuracy, followed closely by AdaBoost, then KNN and Random Forest, as per the outcomes obtained in this study. Implementing the proposed models has an array of benefits to both financial organizations and the US economy in terms of real-time fraud detection, advanced accuracy of fraud detection, cost efficiency, reduction in financial losses as well as strengthening financial organizations.

| KEYWORDS

Financial fraud, Anomaly detection, Naïve Bayes, Multi-layer perceptron, Decision Tree, KNN, SVM, Random forest

| ARTICLE INFORMATION

ACCEPTED: 02 June 2024 **PUBLISHED:** 08 June 2024 **DOI:** 10.32996/jcsts.2024.6.3.1

1. Introduction

According to the *Association of Certified Fraud Examiners*, the estimated cost of financial statement fraud in the U.S. is approximated to be $572 billion per year (Palakurti, 2024). Indisputably, fraud is expensive, and therefore, fraud detection helps save enormous amounts of money by intercepting fraudulent activities before any loss can be experienced. Separate from all these financial implications, fraudulent financial statements are detrimental to the employees, the investors, and the general reliability of corporate financial reports. It disrupts the market efficiency and enhances the transaction cost. Therefore, there is always the need to enhance fraud detection and prevention, given that fraudsters change their techniques with time, security solutions become sophisticated, and blocking fraud has been implemented as part of security modules. As per Ahmed et al. (2020), mitigating fraud is a pivotal part of security systems since fraud detection has become an instrumental step in terms of preventing the harmful implications of fraudulent transactions on service delivery, prices, and a company's reputation. The various traditional fraud detection methods focus on reactive solutions such as reducing losses.

Bakumenko & Elragal (2023), indicate that in the recent past, the demand for machine learning techniques in financial auditing has been relatively high because fraud-related work is complicated to handle manually. Auditing appears at a critical point in the

capital market, and managing fraud-related tasks has been a problem without the assistance of technology, hence calling for increased automation and intelligent solutions. Conventionally, the regulators have adopted rule-based systems to help deal with the vast information they get. Dhanawat (2022), argues that these rule-based systems alert human experts regarding suspicious events provided by static rules that detect possible market anomalies. While instrumental, these rule-based systems still have drawbacks, most notably the large volume of alerts they generate. Therefore, this high volume of data makes it challenging to identify real instances of market manipulation within all the alerts. In that light, Machine learning algorithms could assist address this problem.

1.1 Problem Statement

Retrospectively, traditional methods of fraud detection, which are dependent on predefined rules and human observation, fail to catch up with advanced frauds that are escalating exponentially. These gaps, therefore, call for better fraud detection techniques, able to recognize unusual patterns that precipitate fraud in real-time. Machine learning has proven to be a robust solution by offering techniques that can evolve with new and changing fraud patterns without much manual intervention. Among various machine learning techniques, anomaly detection algorithms are most appropriate because they recognize unusual transactions that deviate from typical behavior (IRJET, 2023). However, different machine learning algorithms differ significantly in their ability to detect fraud in finances; hence, there is an urgent need for their comparative analysis so that the most efficient and effective techniques can be established. The prime objective of this research is to resolve the pertinent issue of financial fraud detection in the USA by performing a comparative study of various machine learning algorithms, particularly concentrating on their anomaly detection capabilities. This research will assess the performance of the machine learning models in terms of accuracy, precision, recall, and computational efficiency, offering valuable insights regarding their practical applicability in real-world financial systems.

2. Related Works

Jidiga & Sammulal (2022), in their study, they analyzed further distorted high-level data on credit card fraud. They aimed to show the analysis of diverse methods, including Naïve Bayes, K-Nearest Neighbor (KNN), and Logistic Regression. In particular, investigational data of European customer transaction records with credit card details were collected, with a total of 284,807 data. In pre-processing the distorted data, the researchers applied a combo strategy in their task for under-sampling and over-sampling. Afterward, the original and pre-processed data was put in the three algorithms. Experiments were explored with Python. The overall results are portrayed below: Naive Bayes, 97.92%; KNN, 97.69%; Logistic Regression, 54.86%. Comparatively, in the task, KNN outperformed the actual best performer between the three: Naive Bayes and Logistic Regression.

Karimi [2023] proposed an Extreme Gradient Boosting-based technique for fraud detection in insurance agreements, followed by the development of a solution for online learning to meet the requirements in real-time. This entailed AI techniques and blockchain architecture which were applied along with the techniques proposed to achieve better security aspects. Machine learning first handled the preprocessing and cleaning of data, followed by data visualization techniques to derive insight. Subsequently, the privacy of the individuals was maintained because no personal identity was supposed to be revealed. Finally, an XG-Boost model was created to predict fraud probabilities based on existing data. An ultra-fast Decision Tree algorithm was created for online learning. Results Comparisons between XG-Boost and other machine learning classifiers, such as the Decision Tree, Naive Bayes, and K-Nearest Neighbour, where the results demonstrated that XG-Boost was relatively more accurate than the latter.

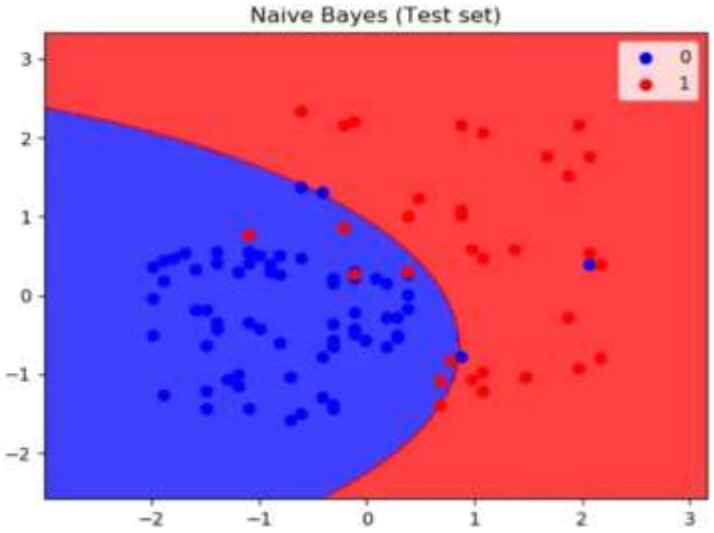
On the other hand, Lokanan et al. [2021] employed a Support Vector Machine (SVM) for the detection of financial frauds and anomalies. Different data mining approaches were also discussed in their research. Data was retrieved from Turkish insurance companies and consisted of the total claimed records and other client information. The grid search resulted in an implemented system with a linear kernel on Oracle. The training was performed by categorizing the records into genuine claims and anomalies. The classification was done by a Support Vector Machine, which compared the individual records with one of the genuine claims against a fake claim. The system, therefore, calculated the probability for every single record, and if the probability was more than 50%, then the record was considered an anomaly. Some of the anomalies to consider included: the total number of rejected claims, the number of uncontrolled claims identified in each type of health center, and the number of allegations identified in health centers. Some methods within the data mining approach that can be applied to determine insurance fraud are grouping, classification, and variance detection.

Morozov [2023], applied various machine learning techniques for predicting and assessing fraud in auto insurance. The prediction and analysis of fraud employed the Bayesian Networks, Decision Trees, and rule-based scheme. The analysts modeled two Bayesian networks, considering the scenario in which the driver might cheat or might not cheat. Afterward, they calculated probabilities independently and chose the output to consider one with a higher probability. The decision trees were based on sub-classes, i.e., the type of account subclassification or label such as "legal" or "fraud". At the same time, the impurity measures within class like Gini, minority, or entropy were used to determine the last release. The rule-based such carried if-then rules where the state was

those conditions facts like driver's age, rating, and the vehicle's age. The results and performance were outlined in the =confusion matrix, which showcased that the accuracy was good enough.

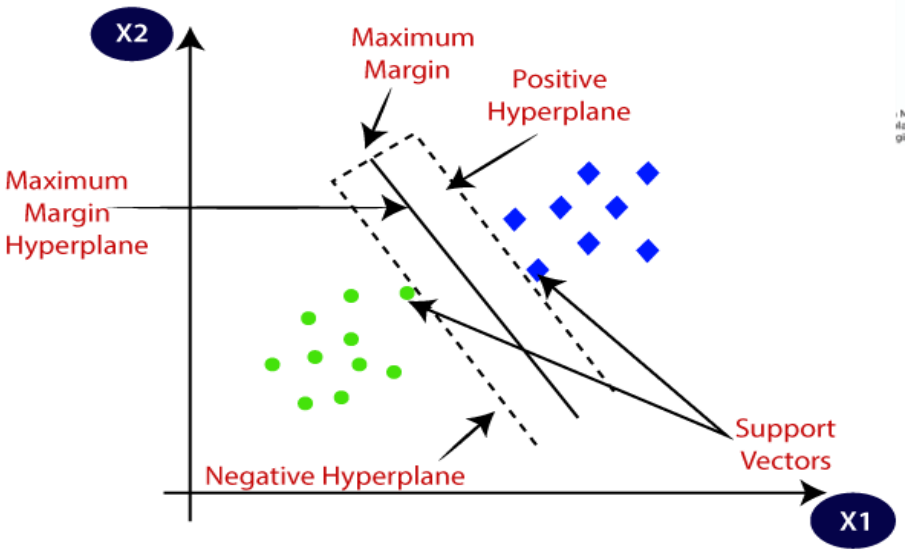
2.1 Types of Machine Learning Algorithms

2.1.1 Naïve Bayes (NB)



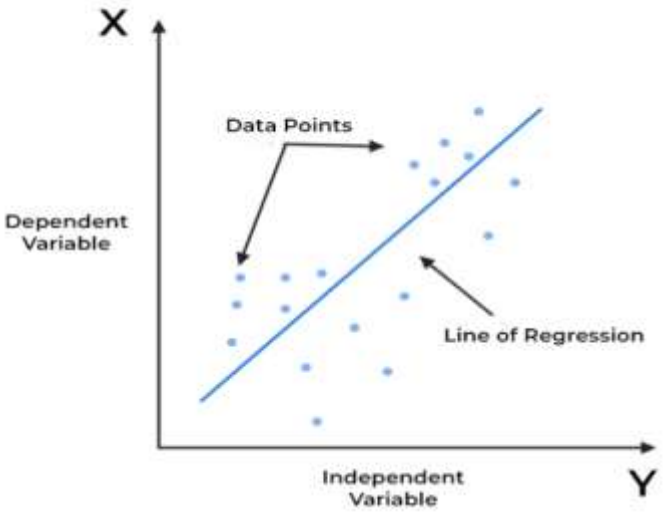
Panda [2024], asserts that Naive Bayes (NB) is a highly renowned algorithm premised on Bayes' Theorem. It computes the probability of instances that belong to a class. It can be considered the most prominent classifier in the world, due to its a priori simplicity, accuracy, and reliability. Although it applies to several domains, NB has seen the most implemented in natural language processing, hybrid recommender systems, text classification, and spam filtering. It is termed "naïve" due to the simplifying assumption that all attributes are independent of each other. By using the prior, the NB calculates the probability for each of the attributes. Even though it assumes this independence among the features, quite a non-realistic assumption, most of the time, it gets it right due to its nature in probabilistic classification.

2.1.2 Support Vector Machines (SVM)



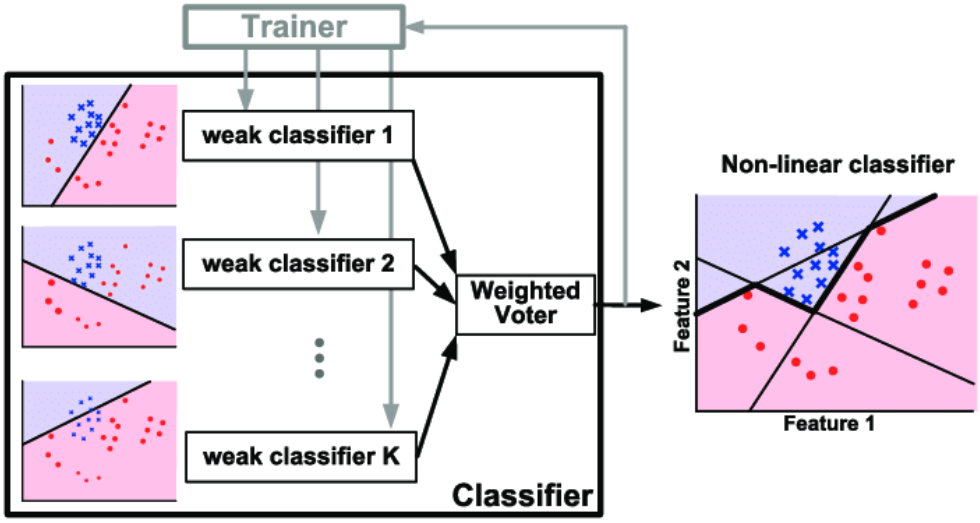
Pooja (2023), articulates that SVM is among the popular machine learning classifiers, and its application is in different domains and spectrum. It can be applied for both linear and nonlinear classification problems, so it can be applied in all real-world domains, as pointed out in reference. The SVM uses a hyperplane that separates the classes of instances. In the case of nonlinear classification, it can use a kernel function to transform feature spaces from low to high dimensions. Thus, the capability of SVM to handle nonlinear relationships through kernel transformations into higher dimensions provides a way of classifying instances in problems that may be very complex themselves.

2.1.3 Linear Regression



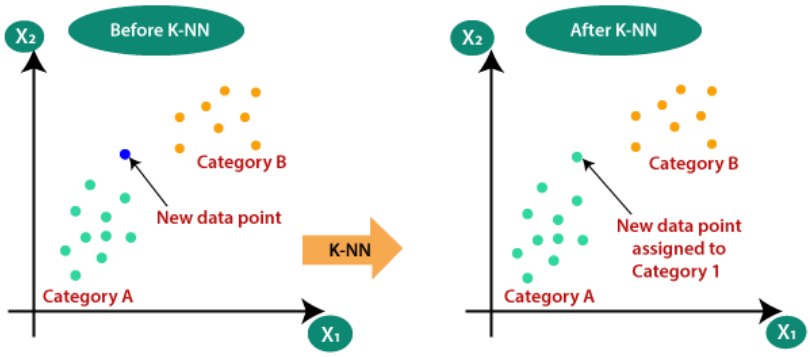
Linear Regression evaluates the association between target and input variables. It is a supervised machine learning algorithm used to predict a continuous numerical, variable, or quantitative response. As opposed to classification, which appoints a class from a group of categories, its output constancy slope. The simplicity of linear regression can be of two types: Simple regression Only one independent variable is used. Multi-variable regression: More than one independent variable is used. The term "linear" bears a linear correlational meaning between variables on the x-axis and the ones on the y-axis[Rane, 2023]. Linear regression effectively models the linear relationship between different sets of variables. Linear regression is used in various activities today because it has proven to model linear relationships between multiple sets; some of these are pricing prediction, trend forecasting, and risk management.

2.1.4 AdaBoost Algorithm



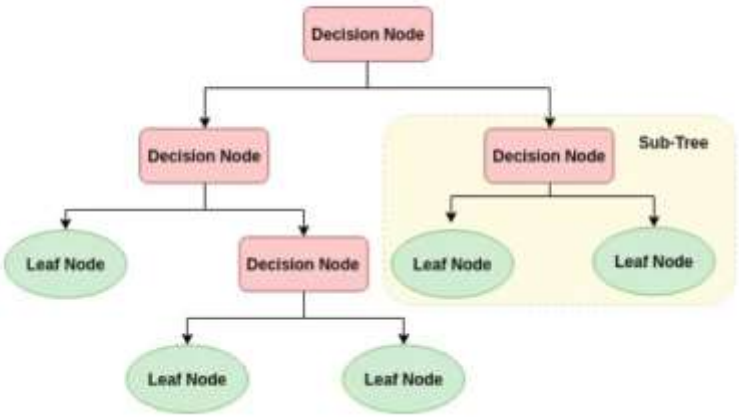
AdaBoost, also deemed as Adaptive Boosting, is an algorithm renowned for its capability to rapidly boost machine learning performance. Boosting model is considered highly efficient at transforming lazy learners into eager learners. The overall objective of AdaBoost is to increase or enhance the power of the weak learners' prediction by training those learners. AdaBoost creates a highly accurate classifier by combining large numbers of modest learners [Zhang, 2022]. It grants all attributes an equal weight at the beginning of the way, but these weights get updated as the algorithm undergoes iterations. As it goes into the iteration process, increasing weight will be given to attributes that have gone wrong under the classification of previous weak learners. In this process of weight adjustment, the focus remains on error minimization and, hence, finally, uses a lot of weak learners to build a robust and eager learner.

2.1.5 K-Nearest Neighbor (KNN)



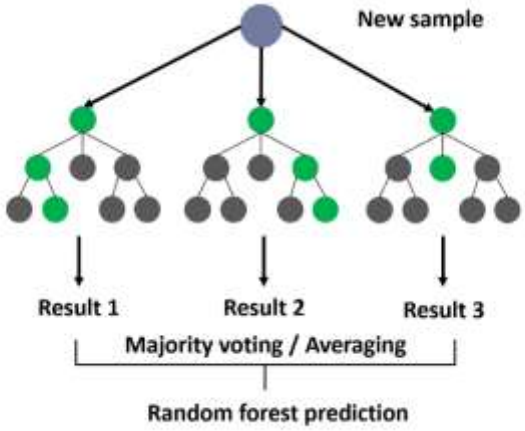
Jidiga & Sammulal (2022), posits that the K Nearest neighbor classifier is mostly employed to classify the scenarios to the neighbor with the majority vote. To find the neighbor with the smallest distance, some distance metric is used. The most common distance measure is the Euclidean distance. The distance is determined among test and training instances. Wherein after the distance is determined, the feature value is calculated for all the nearest neighbor training examples, and the majority of this value is taken as the prediction value based on which the new test dataset is classified. KNN is highly recommended in the scenario where accurate prediction is needed because it is easy to implement.

2.1.6 Decision Trees Algorithm



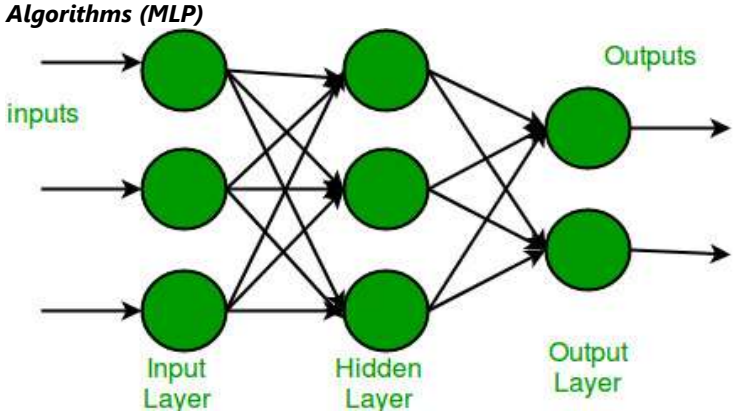
As per Ahmed et al. [2020], decision trees depict results in the form of a tree-like structure. In the decision trees algorithm, inner nodes portray descriptive attributes while leaves are presented with classes. Decision trees are mainly employed in data mining because of their simplicity and robustness. It works by selecting the feature at every classification step that will give rise to the highest gain in information. The classifier stops when all the leaf nodes are pure, which means all the instances in a leaf are of the same class. A leaf node can be pure if further split results somehow in less homogeneity, which suggests that the emergence of the decision tree is perfectly done and no more classification should be done.

2.1.7 Random Forest (RF)



Anandakrishnan et al. [2018], articulates that the Random Forest (RF) classifier was devised by Adèle Cutler and Leo Breiman in 2001. It does its operation via a combination effect of "bagging" and the ability to pick attributes at random while splitting nodes ("subspaces"). Several decision trees are built from a training dataset, and the final prediction decision will be based on a vote's aggregation. RF provides excellent accuracy, and it can handle missing values with optimum efficiency; thus, it is well applicable when data classification is large. It is appropriate for applications dealing with massive Remote Sensing, e-commerce, prediction concerning stock markets, fraud detection, and other interactions involved in detection.

2.1.8 Multilayer Perceptron Algorithms (MLP)



Multilayer Perceptron (MLP) is a form of feedforward artificial neural network (ANN). Artificial neural networks are meant to simulate the human brain, which allows it to receive an input, afterward processing and providing it in ways by which the brain influences an ANN. A perceptron is an essential constituent unit of the ANN, where the output is associated with some weight value accessible and fixed using an activation function [Bakumenko & Elragal, 2023]. ANNs learn representations based on training data so that the desired output variable can be related to them. Since ANNs model complicated patterns in the data, their real-world applications include data compression, character recognition, computer vision, pattern recognition, and robotics. Among the mass of ANN types, the MLP one exists, which may be applied by problems willing to earn on the functionality of a neural network and its learning process.

3. Methodology

This section presents the methodological process of preparing the dataset for experimentation using various machine learning classifiers, such as logistic regression, random forest, Multi-layer perceptron, SVM, Naive Bayes, AdaBoost, decision tree, and KNN. Overall, this research was supposed to compare several different classification algorithms in their performances of fraud detection [Pro-AI-Robikul, 2024]. The primary experimental factor was the classification algorithms. The classification algorithms were applied using Python, an open-source data analysis.

3.1 Data Description

Data transferred for this study was retrieved from Kaggle's website (https://www.kaggle.com/mlg-ulb/creditcardfraud). At the beginning of the experiment, there were 31 variables in the dataset. However, the PCA transformation culminated in the deletion

of some irrelevant attributes because of confidentiality concerns. After PCA transformation, the fields "Time" and "Amount" were kept, and the other attributes were considered irrelevant due to confidentiality issues. Out of, 284,807 transactions contained in the dataset, only 492 of them represent fraud cases [Pro-AI-Robikul, 2024]. Hence, the class imbalance in the label distribution possibly posed a challenge in modeling. Therefore, the analyst, considered the preprocessed data with PCA transformations and anonymization as downloaded from the URL given on the Kaggle website, where the data is publicly hosted for research and modeling on fraud.

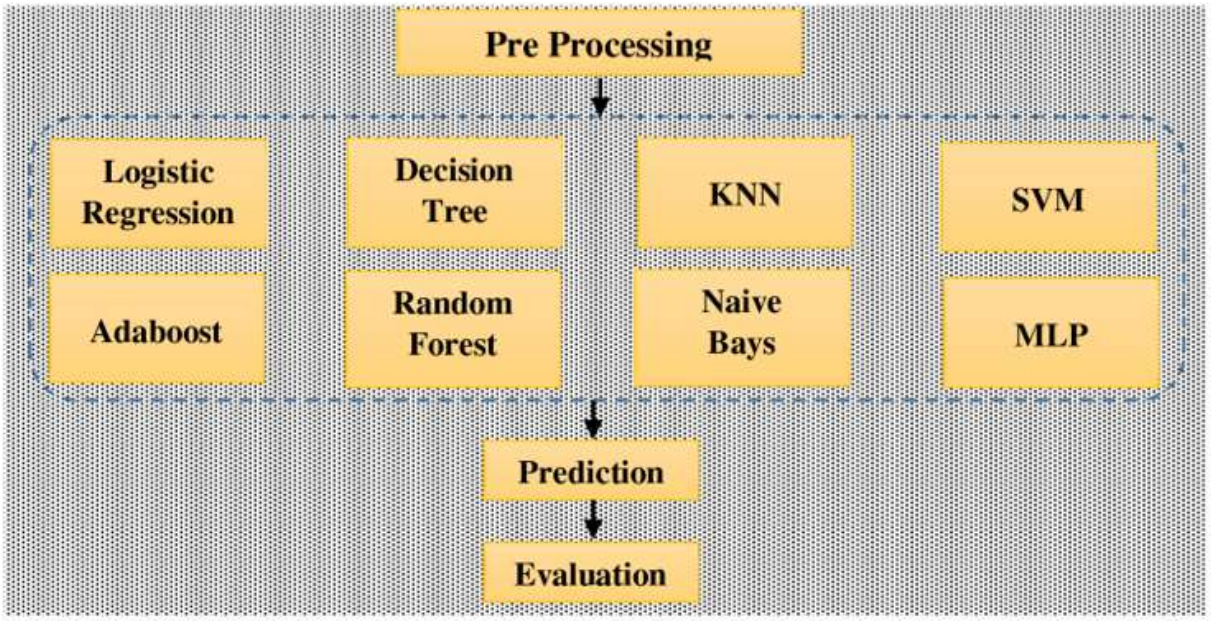
3.2 Feature Engineering and Selection

S/No	Attribute	Description
1.	Amount	Amount of transaction
2.	Event	Event in instants, the occasion in time that has happened since the previous transaction and the first transaction.
3.	Class	1-fraudulent,0-non-fraudulent
4.	V1-V28	Principal component analysis was employed to assess V1-V28 features

3.3 Pre-processing

According to Pro-AI-Robikul [2024], there were four preprocessing procedures conducted before comparing the classification algorithms. Performance was tested upon a 10-fold stratified cross-validation in training with various prior fraud probabilities—0.3%, 0.6%, 1%, 1.5%, 2.5%, 5%, 10%, 20%, 40%, and 60%. Performance is tested via cross-validation, where normalization, discretization, standardization, and continuous predictor filtering are applied. The third step conducted utility testing on the fraud predictors by cross-validation to each classifier. The tuning of the model was executed with cross-validation, fully automated, and executed for 72 hyperparameter-settings configurations. In a way, the preprocessing was done relatively independently without inspecting the different combinations. Ten-fold stratified cross-validation to varying phases of preprocessing in the model training was then used in assessing performance for the best pick out of the amended hyperparameters settings.

3.4 Proposed System



4. Experimentation Results

4.1 Importing Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import IsolationForest
from sklearn.metrics import classification_report, confusion_matrix
```

```
# Load the dataset
data = pd.read_csv("creditcard.csv")

# Display the first few rows of the dataset
print(data.head())
```

Output:

```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         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]
```

In the process of loading the data, further structural transformations were applied to format the data appropriately for each algorithm's input requirements. At the very beginning, the dataset had rows with such attributes as a Transaction ID, time of transaction, and amount of the transaction. Some structural changes were conducted in the loading process to prepare them as inputs that are fed into a classification algorithm with some sort of expectation about the nature or dataset of predictors or features.

```
# Load the dataset
data = pd.read_csv("creditcard.csv")

# Display the first few rows of the dataset
print(data.head())
```

```
   Time        V1        V2        V3        V4        V5        V6        V7  \
0   0.0 -1.359807 -0.072781  2.536347  1.378155 -0.338321  0.462388  0.239599
1   0.0  1.191857  0.266151  0.166480  0.448154  0.060018 -0.082361 -0.078803
2   1.0 -1.358354 -1.340163  1.773209  0.379780 -0.503198  1.800499  0.791461
3   1.0 -0.966272 -0.185226  1.792993 -0.863291 -0.010309  1.247203  0.237609
4   2.0 -1.158233  0.877737  1.548718  0.403034 -0.407193  0.095921  0.592941

         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]
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
```

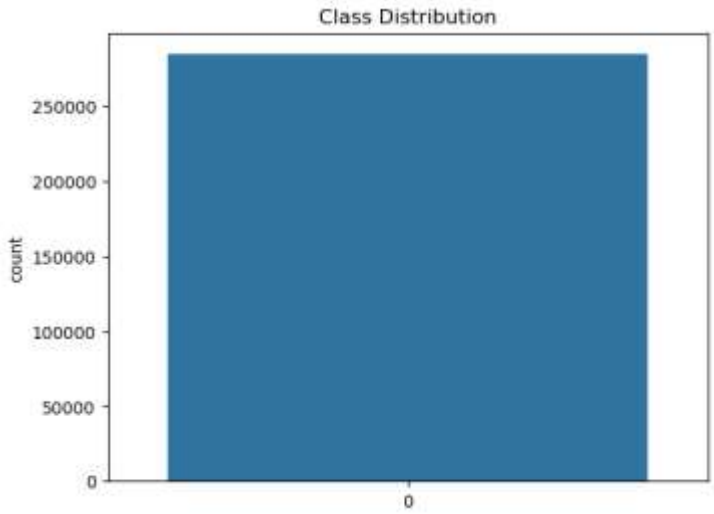
Output:

	Time	V1	V2	V3	V4	V5	V6	V7	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848(
mean	94813.859575	1.168375e-15	3.416908e-16	-1.379537e-15	2.074095e-15	9.604066e-16	1.487313e-15	-5.556467e-16	1.213
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00	1.237094e+00	1.1943
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.321(
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.086
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01	4.010308e-02	2.235
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01	5.704361e-01	3.273
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01	1.205895e+02	2.0007

To visualize the class distribution of the target variables, a suitable code snippet was imposed to generate a histogram, which can be displayed as follows:

```
# Visualize the distribution of the target variable
sns.countplot(data['Class'])
plt.title('Class Distribution')
plt.show()
```

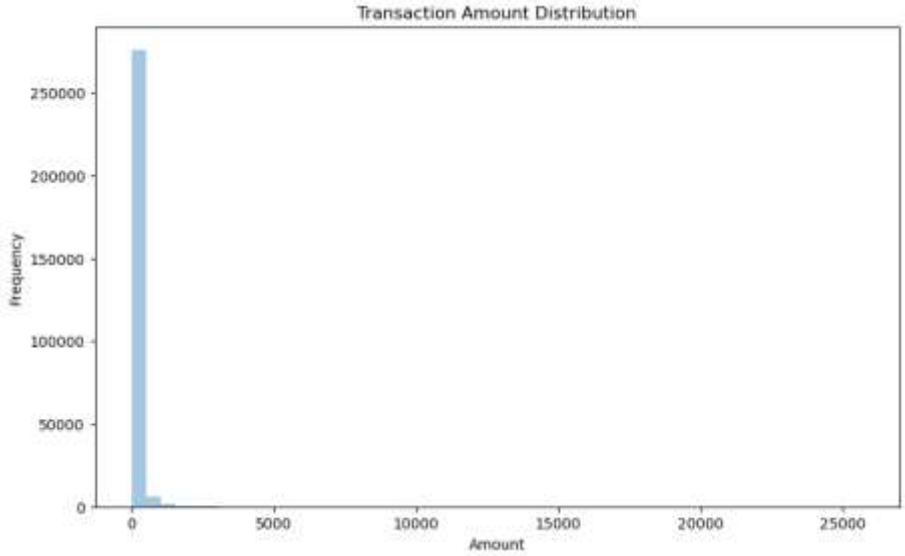
Output:



The data frame was structured so that each row goes to the maximum value of transactions, as recorded from all ATMs, in a particular month; in turn, every row indicated the highest value of such transactions for each month. The data frame was set up this way so that the analyst would be able to see month-on-month, more or less, the trends and patterns in ATM usage across the branches by observation of how the highest monthly transaction amounts compared over the period reflected in the data frame.

```
# Visualize the distribution of transaction amounts
plt.figure(figsize=(10, 6))
sns.distplot(data['Amount'], bins=50, kde=False)
plt.title('Transaction Amount Distribution')
plt.xlabel('Amount')
plt.ylabel('Frequency')
plt.show()
```

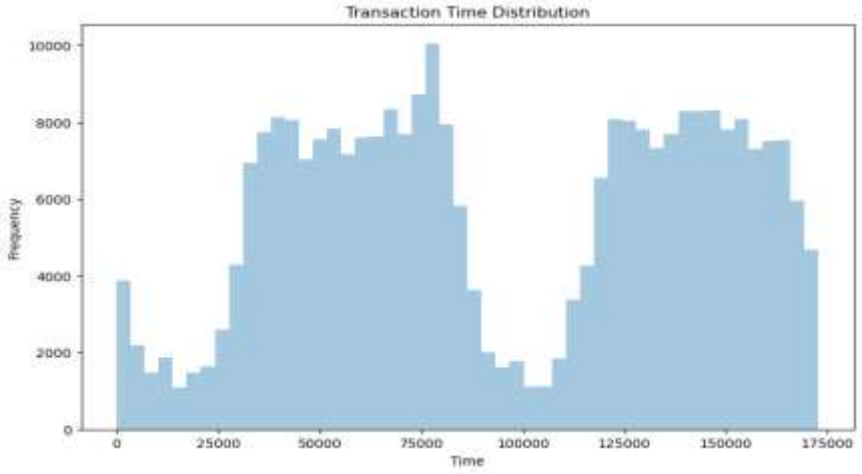
Output:



To display the distribution of transaction times, an appropriate code snippet was imposed. The code snippet represented the transaction date in terms of days and months, therefore providing an extensive comprehension of the period covered by the transaction data.

```
# Visualize the distribution of transaction times
plt.figure(figsize=(10, 6))
sns.distplot(data['Time'], bins=50, kde=False)
plt.title('Transaction Time Distribution')
plt.xlabel('Time')
plt.ylabel('Frequency')
plt.show()
```

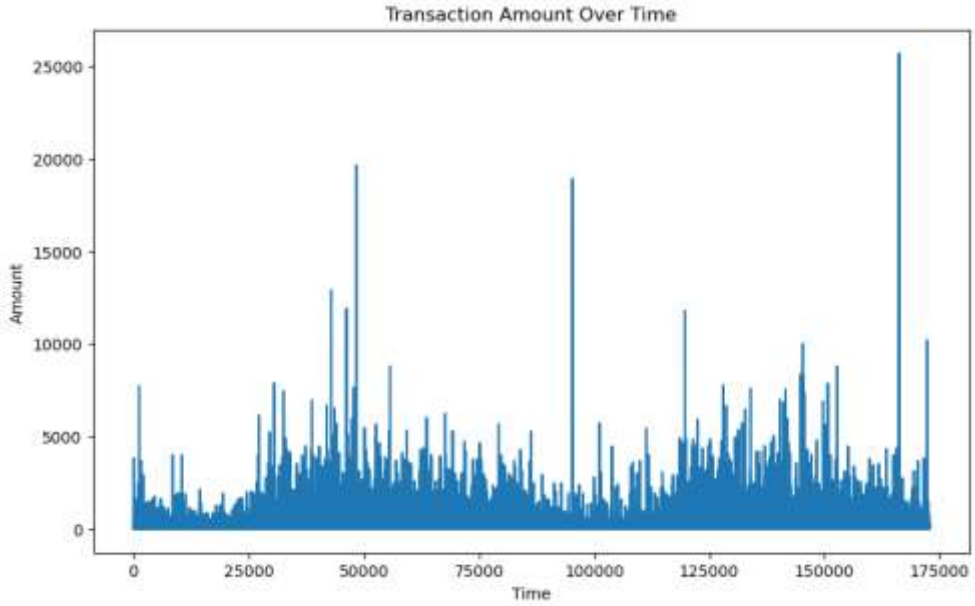
Output:



Furthermore, the analyst was keen to explore the time series of transaction amounts as showcased below:

```
In [15]: # Time Series Plot of Transaction Amount
         plt.figure(figsize=(10, 6))
         plt.plot(data['Time'], data['Amount'])
         plt.title('Transaction Amount Over Time')
         plt.xlabel('Time')
         plt.ylabel('Amount')
         plt.show()
```

Output:



4.2 Performance Measures and Evaluations

The five metrics used for the performance evaluation were accuracy, precision, recall, F1-score, and confusion matrix. Precision computes the proportion of the predicted positive cases that were positive. The recall is considered as the ratio of actual positive cases that were correctly identified as positive. The F1-score is the mean of precision and recall to get a single score. A confusion matrix gives a clear view of correct and incorrect predictions across actual classes, providing a more numerical, broader measure of the machine learning algorithm's performance over a single performance measure. In that respect, Precision, Recall, and F1-Score were computed using the following equations:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+False Positive}}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+False Negative}}$$

$$\text{F1 Score} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}}$$

4.3 Performance Outcomes

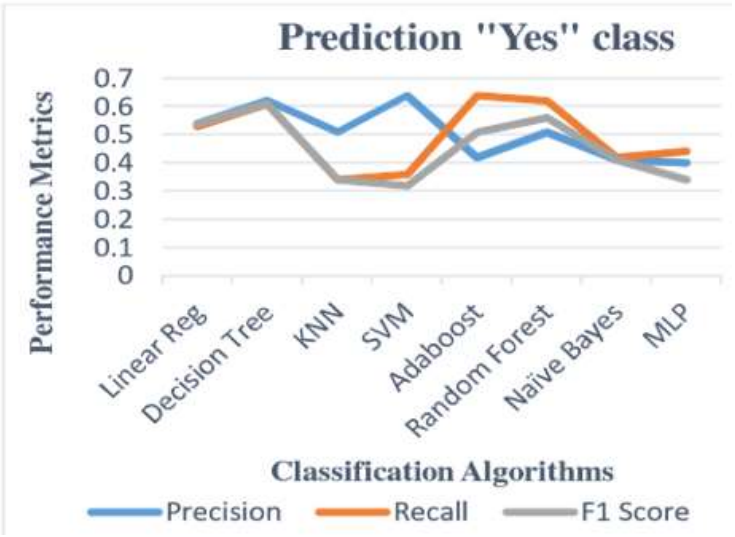


Table 2: Confusion matrix, Accuracy, Precision, Recall, and F1 score

Metrics	Classification Algorithms															
	Linear Regression		Decision Tree		KNN		SVM		Adaboost		Random Forest		Naïve Bayes		MLP	
Confusion Matrix	[[145 55] [0 0]]		[[123 24] [22 31]]		[[145 55] [0 0]]		[[145 55] [0 0]]		[[132 32] [13 23]]		[[126 29] [19 26]]		[[135 51] [0 0]]		[[145 55] [0 0]]	
Precision	0	1.00	0	0.85	0	1.00	0	1.00	0	0.91	0	0.88	0	0.87	0	1.00
	1	0.54	1	0.62	1	0.51	1	0.64	1	0.42	1	0.51	1	0.41	1	0.40
Recall	0	0.72	0	0.85	0	0.72	0	0.72	0	0.80	0	0.83	0	0.81	0	0.72
	1	0.53	1	0.61	1	0.34	1	0.36	1	0.64	1	0.62	1	0.42	1	0.44
F1 Score	0	0.84	0	0.85	0	0.84	0	0.84	0	0.85	0	0.85	0	0.82	0	0.84
	1	0.54	1	0.61	1	0.34	1	0.32	1	0.51	1	0.56	1	0.41	1	0.34
Accuracy	73%		79%		77%		73%		78%		76%		73%		73%	

4.4 Interpretation

The experimental outcomes showcased that the Decision Tree model [79%] performed best overall in terms of robustness to outliers, execution time, and noise reduction, outperforming all the other algorithms tested. By contrast, AdaBoost had almost as high classification accuracy as Decision Tree, with an accuracy of 78%. Conversely, KNN followed at 77% accuracy, and Random Forest obtained 76% accuracy. In summation, Decision Tree had superior performance at classification accuracy, followed closely by AdaBoost, then KNN and Random Forest, as per the outcomes obtained in this study.

4.5 How to Implement the Proposed System

Step 1: Pre-processing US financial organizations should start by gathering a complete dataset of historical transactions. The dataset should include both fraudulent and non-fraudulent transactions for the correct training of data. Subsequently, the analyst should handle missing values, and remove/rectify inconsistency and outliers.

Step 2: Feature Engineering The analyst should then proceed to extract or develop relevant features that may be used to differentiate between fraudulent and genuine transactions. For example, amount, frequency, location, and time of the transaction.

Step 3: Data Splitting- Afterward, the analyst should consider splitting the data into training and testing sets, essentially adhering to an 80-20 or 70-30 ratio, to train algorithms and assess their performance.

Step 4: Model Training-Financial organizations should then train multiple machine learning models such as Logistic Regression, AdaBoost, Decision Tree, KNN, Naïve Bayes, Support Vector Machines, and Multilayer Perception.
Step 5: Hyperparameter Tuning-The analyst should employ methods such as random search or grid search to fine-tune the hyperparameters of each algorithm for optimal performance.
Step 6: Model Application: Subsequently, the researcher should apply the trained algorithms to the test dataset to make a forecast if each transaction is non-fraudulent or fraudulent.
Step 7: Performance Metrics: The analyst should be keen to assess each algorithm utilizing metrics such as precision, accuracy, recall, or F1 score.
Step 8: Model Integration and Deployment At this juncture, the financial organization should consolidate the best-performing algorithm into the company's existing transaction processing system.

5. Business Impact

5.1 Benefits to Businesses in the USA

Advanced Accuracy of Fraud Detection: Multiple machine learning algorithms make the model even more capable of detecting a greater variety of fraudulent activities with high accuracy. For example, it has been proven that Random Forests and Decision Tree can detect patterns of complex fraud—ones that are usually not detectable in rule-based traditional systems.
Real-Time Detection: Machine learning models, once deployed, can perform real-time transaction processing alongside real-time transaction detection activities. This real-time detection feature helps to save fraudulent actions before critical monetary losses can be accrued for both businesses and customers.
Reduction of False Positive: The complex sophistication achieved by machine learning models can differentiate genuine and fake transactions to a much better extent, thus reducing false positive cases.
Adaptability to New Fraud Tactics: Machine learning models are learnable from new observations. In this way, they continuously adapt to new and developing fraud tactics.

5.2 Benefits to the USA Economy

Cost-Efficiency- Machine learning automation of the fraud detection process reduces extensive manual review and intervention, decreasing operational line costs associated with fraud detection that shift human resources into more critical areas, like the investigation of complex cases or improving customer service.
Reduction in Financial Losses-The effective detection of fraud activities scales down financial losses experienced in different financial sectors. Reduction of financial losses implies more capital stays within an economy. In turn, it is reinvested in businesses and consumer spending and further economic growth.
Strengthening of Financial Institutions: Financial institution that efficiently controls fraud are in a better position to maintain financial soundness and stability. Strong and stable financial institutions are vital for a healthy economy.

6. Conclusion

The prime objective of this research was to address the pertinent issue of financial fraud detection in the USA by performing a comparative study of various machine learning algorithms, particularly concentrating on their anomaly detection capabilities. Experimentation was performed using various machine learning classifiers, such as logistic regression, random forest, Multi-layer perceptron, SVM, Naive Bayes, AdaBoost, decision tree, and KNN. Data utilized for this study was retrieved from Kaggle's website (https://www.kaggle.com/mlg-ulb/creditcardfraud). The five metrics used for the performance evaluation were accuracy, precision, recall, F1-score, and confusion matrix. Decision Tree had superior performance at classification accuracy, followed closely by AdaBoost, then KNN and Random Forest, as per the outcomes obtained in this study. Implementing the proposed has an array of benefits to both financial organizations and the US economy in terms of real-time fraud detection, advanced accuracy of fraud detection, cost efficiency, reduction in financial losses as well as strengthening financial organizations.

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Ahmad, M., Ali, M. A., Hasan, M. R., Mobo, F. D., & Rai, S. I. (2024). Geospatial Machine Learning and the Power of Python Programming: Libraries, Tools, Applications, and Plugins. In Ethics, Machine Learning, and Python in Geospatial Analysis (pp. 223-253). IGI Global.

[2] Ahmed, M., Choudhury, N., & Uddin, S. (2017, July). Anomaly detection on big data in financial markets. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017* (pp. 998-1001).

[3] Anandakrishnan, A., Kumar, S., Statnikov, A., Faruquie, T., & Xu, D. (2018, January). Anomaly detection in finance: editors' introduction. In *KDD 2017 Workshop on Anomaly Detection in Finance* (1-7). PMLR.

[4] Anderson, J., & Smith, R. (2024). AI Defenders: Safeguarding the Virtual Gate from Cyber Threats (No. 13297). EasyChair.

[5] Bakumenko, A., & Elragal, A. (2022). Detecting anomalies in financial data using machine learning algorithms. *Systems*, *10*(5), 130.

[6] Dhanawat, V. (2022). Anomaly Detection in Financial Transactions using Machine Learning and Blockchain Technology. *International Journal of Business Management and Visuals, ISSN: 3006-2705*, *5*(1), 34-41.

[7] Gonzalez, S. (2024). Transparency and Trust: Advancing Credit Card Fraud Detection with Explainable AI Models for Enhanced Compliance in the USA. *Innovative Social Sciences Journal, 10*(1), 1-8.

[8] Hasan, M. R., Gazi, M. S., & Gurung, N. (2024). Explainable AI in Credit Card Fraud Detection: Interpretable Models and Transparent Decision-making for Enhanced Trust and Compliance in the USA. *Journal of Computer Science and Technology Studies, 6*(2), 01-12.

[9] IRJET (2021). Comparative analysis of credit card fraud detection using machine learning and deep learning techniques. *Irjet*. https://www.academia.edu/57708314/IRJET_Comparative_Analysis_of_Credit_Card_Fraud_Detection_Using_Machine_Learning_and_Deep_Learning_Techniques?sm=b

[10] Islam, M. Z., Gurung, N., & Gazi, M. S. (2024). Novel AI-Powered Dynamic Inventory Management Algorithm in the USA: Machine Learning Dimension. *Journal of Economics, Finance and Accounting Studies, 6*(2), 156-168.

[11] Jidiga, G. R., & Sammulal, P. (2014, May). Anomaly detection using machine learning with a case study. In 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies (pp. 1060-1065). IEEE.

[12] Kamangar, F. (2024). Decoding Success: The Intersection of Business Analytics and Competitive Advantage. *Social Sciences Spectrum, 3*(1), 126-136.

[13] Karimi, N. (2023). Financial Statement Fraud Detection: An analysis of statistical and machine learning algorithms. *www.academia.edu*. https://www.academia.edu/112490001/Financial_Statement_Fraud_Detection_An_Analysis_of_Statistical_and_Machine_Learning_Algorithms?sm=b

[14] Liu, C., & Anderson, J. (2024). Beyond Firewalls: AI's Evolutionary Role in Cybersecurity (No. 13304). EasyChair.

[15] Lokanan, M., Tran, V., & Vuong, N. H. (2019). Detecting anomalies in financial statements using machine learning algorithm: The case of Vietnamese listed firms. *Asian Journal of Accounting Research*, *4*(2), 181-201.

[16] Mafiqul Islam, M. (2024). Artificial Intelligence Exploring Its Applications across Industries. *Journal of Artificial Intelligence General science* (JAIGS) ISSN: 3006-4023, 2(1), 20-24.

[17] Morozov, I. (2016). *Anomaly detection in financial data by using machine learning methods* (Doctoral dissertation, Hochschule für Angewandte Wissenschaften Hamburg).

[18] Palakurti, N. R. (2024). Challenges and Future Directions in Anomaly Detection. In *Practical Applications of Data Processing, Algorithms, and Modeling* (pp. 269-284). IGI Global.

[19] Panda, K. (2024). Anomaly Detection of Financial Data using Machine Learning. *www.academia.edu*. https://www.academia.edu/117366704/Anomaly_Detection_of_Financial_Data_using_Machine_Learning?sm=b

[20] Pooja, S. (2023). A comparative study of credit card fraud detection using machine learning. *www.academia.edu*. https://www.academia.edu/100983250/A_comparative_study_of_credit_card_fraud_detection_using_machine_learning?sm=b

[21] Pro-AI-Rokibul. (2024). *Anomaly-Detection-On-Financial-Fraud/Model/Anomaly Detection. ipynb at main · proAIrokibul/Anomaly-Detection-On-Financial-Fraud*. GitHub. https://github.com/proAIrokibul/Anomaly-Detection-On-Financial-Fraud/blob/main/Model/Anomaly%20Detection.ipynb

[22] Rane, K. (2023). Financial fraud detection in healthcare using machine learning and deep learning techniques. *www.academia.edu*. https://www.academia.edu/55969281/Financial_Fraud_Detection_in_Healthcare_Using_Machine_Learning_and_Deep_Learning_Techniques?sm=b

[23] Troisi, O., & Maione, G. (2024). Data-Driven Decision Making: Empowering Businesses through Advanced Analytics and Machine Learning. *Journal Environmental Sciences And Technology, 3*(1), 515-525.

[24] Vassakis, K., & Petrakis, E. (2024). Mastering the Data Universe: Leveraging Big Data for Competitive Advantage and Market Domination. *Journal Environmental Sciences And Technology, 3*(1), 526-236.

[25] Zhang, Q. (2022). Financial data anomaly detection method based on decision tree and random forest algorithm. *Journal of Mathematics*, *2022*(1), 9135117.