Internet of Things and Cyber-Physical Systems 4 (2024) 293–306

Contents lists available at ScienceDirect

Internet of Things and Cyber-Physical Systems

journal homepage: www.keaipublishing.com/en/journals/internet-of-things-and-cyber-physical-systems

Credit card default prediction using ML and DL techniques

Fazal Wahab [a,*,1], Imran Khan [b,**,1], Sneha Sabada [c,1]

[a] Corvit Systems Peshawar, Pakistan
[b] Department of Informatics, University of Sussex, UK
[c] Department of Engineering and Computer Science, University of Hertfordshire, UK

ARTICLE INFO

Keywords:
Deep learning
Machine learning
Credit card default prediction
Ada boost
Decision tree

ABSTRACT

The banking sector is widely acknowledged for its intrinsic unpredictability and susceptibility to risk. Bank loans have emerged as one of the most recent services offered over the past several decades. Banks typically serve as intermediaries for loans, investments, short-term loans, and other types of credit. The usage of credit cards is experiencing a steady increase, thereby leading to a rise in the default rate that banks encounter. Although there has been much research investigating the efficacy of conventional Machine Learning (ML) models, there has been relatively less emphasis on Deep Learning (DL) techniques. The application of DL approaches to credit card default prediction has not been extensively researched despite their considerable potential in numerous fields. Moreover, the current literature frequently lacks particular information regarding the DL structures, hyperparameters, and optimization techniques employed. To predict credit card default, this study evaluates the efficacy of a DL model and compares it to other ML models, such as Decision Tree (DT) and Adaboost. The objective of this research is to identify the specific DL parameters that contribute to the observed enhancements in the accuracy of credit card default prediction. This research makes use of the UCI ML repository to access the credit card defaulted customer dataset. Subsequently, various techniques are employed to preprocess the unprocessed data and visually present the outcomes through the use of exploratory data analysis (EDA). Furthermore, the algorithms are hypertuned to evaluate the enhancement in prediction. We used standard evaluation metrics to evaluate all the models. The evaluation indicates that the AdaBoost and DT exhibit the highest accuracy rate of 82 % in predicting credit card default, surpassing the accuracy of the ANN model, which is 78 %.

1. Introduction

The financial markets are paying increasing attention to a borrower's potential for loan repayment. There are a number of considerations reviewed before a loan is approved. The elements considered in the evaluation process include the applicant's present financial situation, the requested amount, the age of the applicant, the profession, and other relevant factors [1]. As credit cards are classified as a type of credit, the monthly payment amount can fluctuate. Pre-approval determines the upper limit of a consumer's spending capacity. The prevalence of credit card usage for daily purchases at both physical and virtual retail establishments has resulted in a massive rise in the use of credit card numbers and debt in recent years. Almost all banks are required to deal with instances of credit card fraud, which might involve activities such as disposal and defaults [2]. Furthermore, due to the existence of numerous deceptive strategies, regular retraining is necessary to effectively identify any instances of credit card fraud. The task of preventing credit card fraud is further complicated by the disparity between the expenses associated with declining a valid card and those associated with providing legal cards.

In today's digital environment, it's common practice for banking sectors and other monetary institutions to develop detection mechanisms for fraud that are customized for their client accounts. By assessing regular and aberrant behaviour patterns and individual transactions, Machine Learning (ML) and data mining are two methods that are gaining popularity for use in the process of spotting probable instances of fraudulent conduct. Considering the above circumstances, the most pragmatic and cost-effective approach would be to utilize statistical algorithms to extract potential evidence of fraud from the available data. The objective of supervised models, which are trained on labelled data, is

* Corresponding author.
** Corresponding author.
E-mail addresses: fazalwahabstu@gmail.com (F. Wahab), imran.khan@sussex.ac.uk (I. Khan), sneha.sabada@herts.ac.uk (S. Sabada).
[1] Contributing authors.

https://doi.org/10.1016/j.iotcps.2024.09.001
Received 4 June 2024; Received in revised form 24 July 2024; Accepted 3 September 2024
Available online 6 September 2024

to provide a fraud probability value for individual transactions. This score is determined through a statistical evaluation of the characteristics exhibited by typical fraudulent transactions [3]. Out of the several supervised algorithms, the neural network stands out as a notable solution. Decision Tree (DT), Support Vector Machines (SVMs), and various approaches have also been employed.

The advent of DL innovation has fundamentally transformed how corporations and consumers handle and predict their financial choices [4]. DL systems excel in accurately and rapidly predicting credit card defaults by efficiently processing vast quantities of data. DL can enhance the accuracy of credit card default forecasts when compared to traditional methods such as AdaBoost and DT. DL algorithms have the capability to discern intricate patterns in credit card default patterns due to their ability to comprehend minor correlations and patterns inside extensive datasets. This may lead to improved risk assessment and more accurate forecasts. DL can extract relevant features from raw data, which leads to a higher prediction accuracy compared to conventional approaches [5]. Consequently, DL has become an essential element of the credit card sector, with numerous institutions depending on DL models to precisely forecast client action [6].

Credit card default occurs when consumers fail to fulfill the minimal repayment for their credit card bill consecutively for two or more reporting cycles [7]. These actions can result in adverse outcomes for the cardholder, such as incurring late fees, fines, detrimental impact on their credit rating, and other legal actions from the credit card issuing authority. The banking sector currently utilizes ML techniques, particularly those that apply diverse categorization algorithms, to accurately categorize consumers into distinct categories for improved trend predictions. One benefit of utilising ML for credit card default predictions, as opposed to traditional methods, is its ability to accurately detect probable default risks. The accuracy of credit card default prediction models is contingent upon the reliability and quality of the input data used for training them [2]. Insufficient or inadequately processed data might lead to inaccurate projections. This study investigates the application of DL for credit card default predictions, as DL is capable of capturing intricate patterns in data that conventional ML algorithms cannot. Additionally, DL can analyze substantial volumes of data within a brief timeframe. DL is a highly effective method for credit card companies to minimize the probability of defaults and the associated expenses.

This research aims to find out which DL parameters directly contribute to more precise forecasts of credit card defaults by comparing them to the accuracy of Adaboost and decision tree approaches. For this study, the dataset is downloaded from Kaggle and contains information about credit card defaults. Some examples of the types of information that will be included in this dataset are limitations on credit, payment histories, and geographic data. The preprocessing procedure will be applied to the dataset to get rid of duplicates and eliminate any missing data before the actual analysis begins. To rank the performance of the models, we will use different standard evaluation metrics including recall, precision, accuracy, and F1-Score. The impact that different hyperparameters have on the performance of DL algorithms will also be investigated. The results of this study will improve the accuracy of credit card default prediction for banking organizations and other businesses. The study will also provide information on the role that DL can play in enhancing the precision with which credit card default predictions can be made. Furthermore, we will compare the accuracy ratings and the execution times of the models to find the most accurate model for forecasting credit card defaults.

2. Related work

When a cardholder misses their minimum payment for two or more consecutive billing cycles, they are said to be in credit card default. These actions can result in adverse outcomes for the cardholder, such as incurring late fees, penalties, detrimental impact on their credit rating, and the possibility of facing legal repercussions from their credit card provider [8]. Credit card default predictions have been achieved through numerous ML and DL methods, including DT, LR, and Neural Networks (NN). There are benefits and drawbacks to each possible model, and the choice of model depends on aspects such as the availability of data, the predictive capability of the model, and the training time required [9]. The banking company has grown more dependent on analytical techniques, including predictive analytics, to detect credit card defaults and other possible hazards in recent years. Predictive analytics, as a means of predicting future events, analyze historical data to identify potential trends or patterns that can be leveraged [10]. Predictive analytics has empowered banks to gain a deeper comprehension of their consumers, foresee potential problems, and proactively implement measures to enhance the customer experience and diminish default rates.

Credit card default data imbalance was investigated at a financial institution in Ref. [11]. To remedy this, BPnn and k-means SMOTE were incorporated into a prediction model, leading to improved prediction accuracy. After applying the k-means SMOTE method to alter the data distribution, a BPnn is used to make predictions. The RF method is used to calculate the relevance of data features, and the results are substituted for the default values in the BPnn's weights. The model offered a practical approach to addressing the issue of non-proportional sampling. In addition, this research constructs and compares the performance of five popular ML models (KNN, LR, RF, SVM, and DT). As shown by the experiments, the proposed method improves the model's AUC from 0.765 to 0.929. According to the findings of a study [12], credit card default was accurately predicted using transaction flow data by giving the highest weight to parameters associated with transaction flow. They compared XGBoost-based financial classification models with Long Short-Term Memory (LSTM)-based time-series information models. The XGBoost model's accuracy is based on the efficiency of feature extraction, in contrast to the LSTM technique, which can achieve high accuracy even without feature extraction. The XGBoost-LSTM model did exceptionally well when tasked to categorize default predictions.

The authors of [13] reveal a variety of possible algorithms for determining the legitimacy of a transaction. Data from the Credit Card Fraud Detection database was used for the study. Oversampling via the SMOTE method was employed due to the significant imbalance in the dataset. In their experiment, they used RF, LR, NB, and MLP algorithms. The outcome of this study presents that the RF framework can be applied to detect fraudulent credit card charges with a precision of 99.96 %.

The authors [14] proposed a new approach to identify fraudulent activities. The strategy involves categorising clients into groups according to their spending patterns and extracting behavioural traits to establish a unique identity for each cardholder. The approach analyzed historical transaction data of customers to derive discernible behavioural patterns. The sliding window approach is utilized to aggregate the transactions conducted by cards from different categories. Subsequently, various classifiers, including isolation forest, SVM, local outlier factor, RF, LR, and DT are individually trained on the groups. It was determined that RF, LR, and DT produced the best results. In this study [15], the authors aimed to determine the feasibility of real-time detection of credit card fraud. To accomplish this, they utilize predictive analytics performed by the built ML models, such as determining whether or not a transaction is legitimate. To employ LR, NB, LR, and SVM, they integrated an API module. In addition, the authors assess a novel approach to dealing with the asymmetrical nature of the data. With an accuracy rate of 91 %, the SVM is the most accurate ML model for identifying the fraudulent use of credit card events.

The author's [16] research shows that by modelling prior transactions with data gained from fraudulent ones, credit card theft can be detected. This paradigm can therefore be used to determine whether or not a recently completed transaction was legitimate. Every fraudulent transaction should be identified with as few false positives as possible. Among these methods are the Isolation Forest algorithm and the Local Outlier Factor. Even with almost 99.6 % accuracy considering all available data, the algorithm is only 28 % accurate. When the complete dataset is

utilized, however, accuracy increases to 33 %. Given the huge discrepancy between the valid and authentic transaction numbers, this level of accuracy is to be expected. An effective strategy proposed by Ref. [17] to mitigate the financial losses caused by illicit activities involves the utilisation of data mining and ML techniques to combat fraudulent operations. An ML algorithm was employed to automatically predict the likelihood of transactions being suspicious or non-suspicious, utilising a classifier. The successful combination of ML and data mining technologies enabled the identification of patterns that distinguish authentic transactions from fraudulent ones in the provided data. This paper examines the application of various classifiers, such as TAN, logistics, K2, NB, and J48, in the context of supervised classification. Following the preprocessing steps of normalisation and PCA, all Bayesian classifiers had an accuracy of over 95.0 %.

Another study conducted by Ref. [18] implemented RF and Adaboost ML techniques in the development of a credit card fraud detection system. The Authors assessed the performance of RF and Adaboost algorithms in fraud detection and selected the method that achieved the maximum precision, accuracy, recall, and F1-Score. No substantial disparity in performance was observed between the RF and Adaboost approaches. The findings have led them to the determination that RF outperforms Adaboost in identifying fraudulent transactions on credit cards. In the study conducted by Ref. [19], three different methods, namely DT, LR, and RF, were examined for their effectiveness in detecting credit card fraud. The authors conducted oversampling to enhance the representativeness of the data set, resulting in the identification of 60 % fraudulent transactions and 40 % real transactions The R programming language is utilized for dataset analysis, employing three distinct methodologies. The accuracy levels exhibited by the DT, LR, and RF classifiers are 90.0 %, 94.3 %, and 95.5 %, respectively. Based on the results of the comparison, the RF technique demonstrates significantly superior performance.

The authors of the paper [20] presented a robust method for identifying fraudulent credit card transactions. This method utilizes a feedback loop and relies on ML technology. Subsequently, the authors examined the performance of different techniques, including SVM, NB, LR, ANN, RF, KNN, and GBM classifier algorithms, on a credit card fraud dataset that was slightly imbalanced. The dataset consisted of transaction data acquired from European account holders who conducted a total of 284, 807 credit card transactions. The findings indicate that RF methods demonstrate an accuracy rate of 95.988 %.

The aforementioned extensive literature survey reveals a notable research deficiency in the domain of credit card default prediction with regard to the application of DL techniques. Although there has been much research investigating the efficacy of conventional ML models like LLR, DT, RF, and SVM, there has been relatively less emphasis on DL techniques. The application of these DL approaches to credit card default prediction has not been extensively researched despite their considerable potential in numerous fields. Moreover, the studies frequently lack particular information regarding the DL structures, hyperparameters, and optimization techniques employed, hence impeding a comprehensive assessment of the efficacy of these factors. Furthermore, the preceding research does not extensively investigate the performance comparison between DL algorithms and ML algorithms. By addressing this research gap, we can learn more about the possibilities of DL methods and the effect of factors on credit card default prediction. In this research, we compare the efficiency of various approaches to detecting credit card fraud using DL algorithms with varying hyperparameters. The effectiveness of ML algorithms is then measured against it.

The importance of this research lies in its ability to offer valuable insights into DL algorithms and the most effective mix of hyperparameters for achieving maximum accuracy and minimising false positives in credit card fraud detection. This study will also demonstrate the significance of data preparation approaches and their impact on different models in detecting credit card fraud. Furthermore, a thorough evaluation of the performance of ML techniques in domains such as credit card fraud detection will be conducted. This paper can additionally aid in identifying any potential vulnerabilities of ML and DL techniques in this specific application and determining strategies to mitigate them.

3. Methodology

DL enables computers to autonomously and directly extract valuable representations and characteristics from unprocessed data. This enables them to circumvent the laborious and time-consuming effort of manually generating the representations and characteristics. The prevalence of DL models is primarily attributed to the extensive utilisation of ANN in their diverse manifestations [21]. DL use optimizers to fine-tune several parameters of a neural network, including its weights and learning rates. Hence, it assists in reducing overall loss and increasing overall precision [22]. The activation function alters the input in a manner that is not linear, enabling the system to acquire knowledge and perform increasingly complex tasks. ML employs two methods, namely the DT and the Adaboost classifier, to construct a precise model for predicting credit card defaults. DT are useful for categorising items by their learned characteristics and addressing classification problems. Furthermore, it is employed not only for addressing regression problems but also for forecasting continuous outcomes with unanticipated data [23]. AdaBoost operates by assigning greater weight to cases that are challenging to differentiate and assigning lower weight to instances that have already been well handled. AdaBoost's algorithmic method is applicable to both regression and classification problems [24].

Three distinct models have been used in this study: Adaboost, DT, and ANN. The effectiveness of each model has been evaluated using F1-score, accuracy, precision, and recall. Different hyperparameter configurations, including 3, 4, and 5 hidden layers, optimizers including Adamax, Rmsprop, Adam, SGD, and activation functions including Sigmoid, Relu, Elu, and Tanh, are utilized to further fine-tune the DL model. Finally, a comparison is made between the 3 models' outputs to ascertain which model provides the most precise prediction of credit card defaults. We executed the implementation on a 7G laptop equipped with a 256 GB SSD, 32 GB of RAM, and a 1 TB hard drive. Python is used as a coding language and the Anaconda Data Science Platform with Jupyter Notebook was utilized. The details of the Implementation Procedure is given in Table 1.

The fundamental purpose of this article is to find out how well DL can predict credit card defaults in comparison to other ML approaches like Adaboost and DT, and what values for the corresponding DL parameters produce the most accurate forecasts. ML and DL algorithms, including ANN, Adaboost, and DT, are employed in this study. Recall, accuracy, precision, and the F1-score are used to rank these methods. The impact of different hyperparameters on the precision of DL algorithms is also investigated. This study's experiment also provides information on how DL can be utilized to boost the precision with which credit card default predictions can be made. The experiments performed for this study are

Table 1
Implementation procedure.

Implementation Steps	Explanation
Dataset	Credit Card default Clients Dataset from Kaggle having 25 columns and 30000 rows
Preprocessing	The dataset contains 35 duplicates that have been removed. No null values in the dataset Removed unwanted column 'ID'.
EDA	Joint plot Count plot Hist plot.
Utilized Algorithms	DL Model: ANN Hypertuned by varying its Hidden layers, Activations, Optimizers, ML Models: AdaBoost and DT
Models Evaluation	Precision, Accuracy, F1-Score and Recall

summarised in Figure 1.

3.1. Dataset description and preprocessing

The following URL is used for the dataset download.
https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset.

Before being made available on Kaggle, this dataset was stored at the UCI ML repository. Data from 30000 Taiwanese credit card holders who used their cards between April 2005 and September 2005, including details about their demographics, default payment records, and credit histories. Credit card users who were either overdue on payments or didn't pay at all are shown in this data set. Client IDs, limits on credit, sex, level of education, marriage status, ages, the balance due on the billing statement, payment amount, client demographics, credit information, history of payments, and monthly bills are just some of the 25 unique features included in this dataset. Whether or not the customer has already missed their next payment is likewise a goal variable included. The first steps in developing a reliable credit card default prediction model are collecting and preparing relevant data. Categorical variables are converted to numeric variables, and missing values, duplicates, and unnecessary columns are removed during the preprocessing stage of the dataset.

3.1.1. EDA for credit card default prediction

Exploratory Data Analysis (EDA) helps find credit card customer factors that predict default. The dataset can be analyzed and visualized to uncover correlations that can predict credit card payment defaults. To analyze feature distribution, count plots, histograms, and joint plots are created. Figure 2 represents the count plot for 'default.payment.next.month'.

Figure 3 shows the count plot for 'marriage'. The percentage of late cardholders (default.payment.next.month = 1) against on-time payers (default.payment.next.month = 0) is shown in the countplot. As can be seen, only a small fraction of credit card holders actually end up in

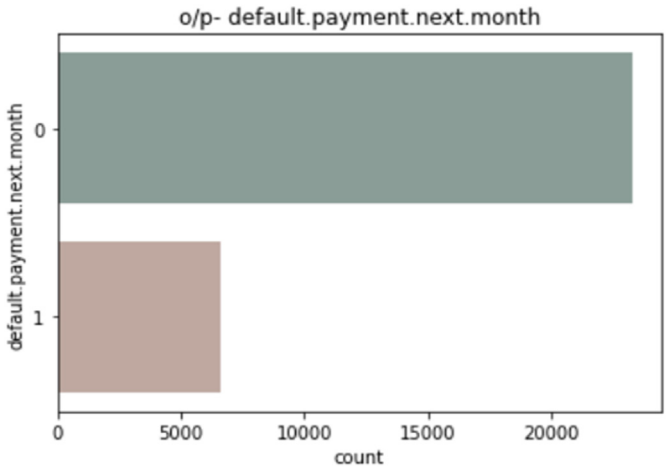

Fig. 2. The count plot for 'default.payment.next.month'.

arrears. A count plot of the attribute "SEX with MARRIAGE" shows that more women than men have credit cards. Additionally, it discloses that the number of married males utilising credit cards exceeds the number of single men, whereas the number of single women using credit cards surpasses the number of married women. Figure 4 shows the joint plot for 'Bill_AMT5'.

Figure 5 is the histogram of 'Bill_AMT6'. According to the histogram of 'BILL_AMTS' shown above, 'BILL_AMT' can contain more than 25000 observations. The combined graph shows how BILL_AMT6 and PAY_AMT6 are related to one another. There is a positive association between the BILL_AMT6 and the PAY_AMT6 because as the former increases, the latter does as well. Nevertheless, there are sporadic observations that deviate from this pattern; thus, the relationship is not perfect. It also stands for the highest frequencies, which fall between 0.0 and 0.4. The new 'csv' format file is used to save the data before it is used

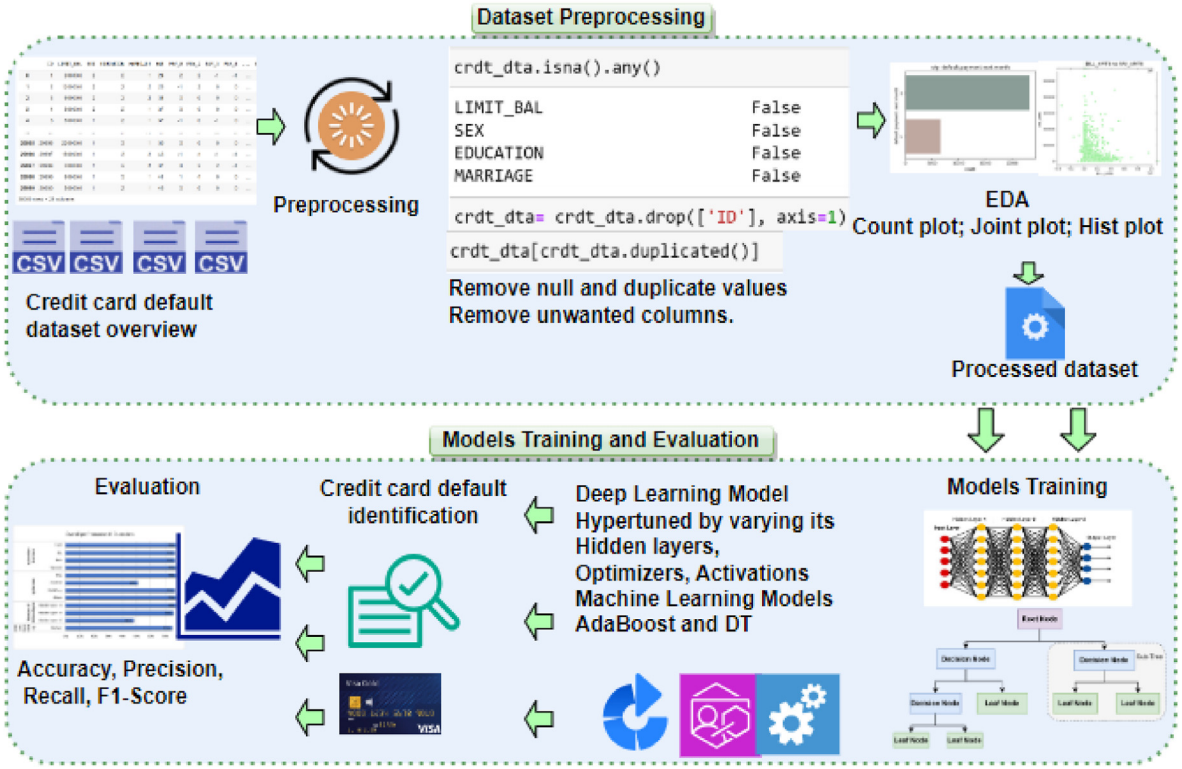

Fig. 1. Research methodology.

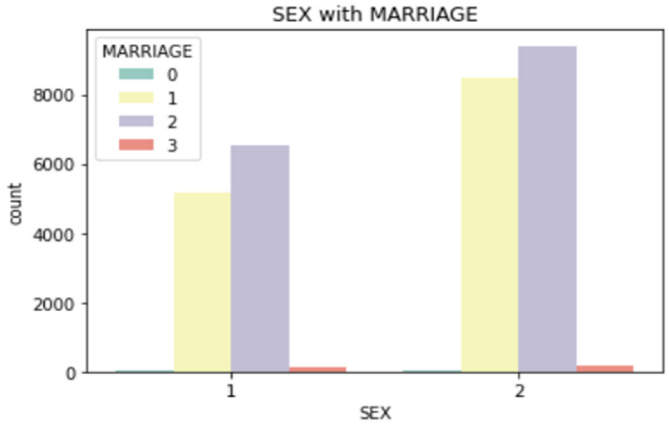

Fig. 3. Count plot for 'marriage'.

in the system.

3.1.2. Dataset dvision

The cleaned dataset has 29965 rows and 24 columns, where 0 has 23335 data points and 1 has 6630 data points. One of the crucial processes in creating ML and DL models is partitioning the dataset. The dataset is divided into three subsets: a validation set, a training set, and a test set. Subsets of the dataset can be used to train and test the model to make sure it is correctly generalizing and producing accurate predictions.

3.2. Deep Learning models

DL builds upon classical ML by enhancing the model's depth and transforming the data through the utilisation of diverse functions. This makes it possible to express the data in a hierarchical fashion using multiple layers of abstraction. One of the primary benefits of DL is the ability to highlight. The process of automatically extracting features from unprocessed data is called "learning to feature". Higher levels of the hierarchy's attributes are produced by combining traits from previous levels [25].

The complex models employed in DL have the potential to enhance classification accuracy or reduce regression errors, provided that there are ample large datasets that accurately represent the situation at hand. The network architecture encompasses the specialized elements of DL, such as pooling layers, convolutions, fully connected layers, encoding/

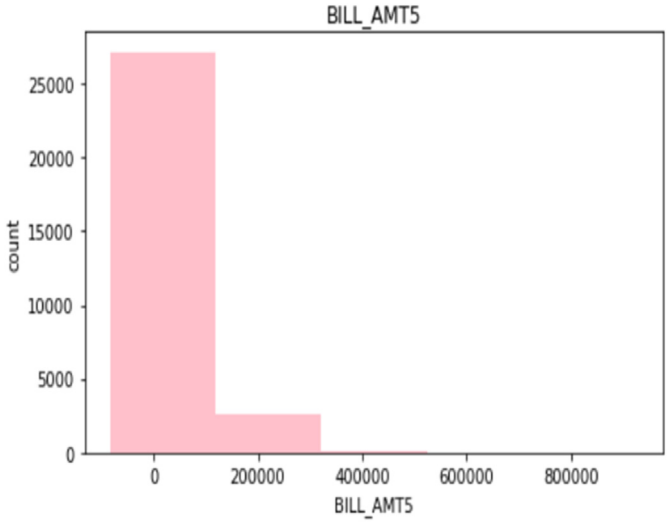

Fig. 4. Histogram for 'BILL_AMTS'.

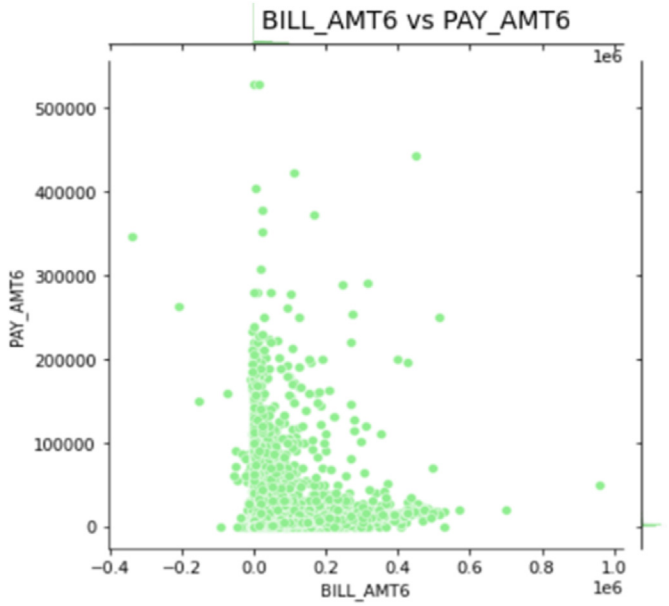

Fig. 5. Joint plot for 'BILL_AMT6' & PAY_AMT6.

decoding methods, memory cells, activation functions, and gates. DL models are highly effective in performing classification and prediction tasks due to their extensive learning capabilities and hierarchical architecture, rendering them well-suited for a wide range of complicated issues [26]. DL can be applied to several types of data, including speech, natural language, audio, continuous data (such as meteorological information), and point data (such as soil chemistry and demographic data). DL has been extensively utilized in several applications that include raster-based data, such as videos and images. However, it is also applicable to any other sort of data [27]. The following figure is the source citation of [28]. The DNN structure is shown is Figure 6.

DL exhibits exceptional capability in managing a vast array of attributes, hence enhancing its computational ability and flexibility when employed on unorganized data. The DL algorithm processes the data in a hierarchical manner, where each subsequent layer can extract a greater quantity of information before passing it to the next layer [29]. The first layer's job is to extract low-level features; the next layer's job is to combine higher-level attributes to create a complete representation. Classifiers learned using DL have much superior performance when trained with larger datasets compared to classifiers developed using traditional approaches [30]. Unlike DL, standard ML methods demonstrate a stable performance once they surpass a specific threshold of training data. Google uses picture and voice recognition, Netflix and Amazon employ recommendation engines, Apple's Siri provides assistance, auto-replies to emails and texts are automated, and chatbots are utilized in various applications. These are only a few instances where DL is now being utilized [31].

3.2.1. Hyper tuning parameters

Optimizer: A method called a neural network optimizer modifies network configurations to improve efficiency. As a result, it helps with improved aim and decreased adverse effects. Choosing appropriate model weights is a difficult task. An optimizer is a kind of DL algorithm that adjusts a model's parameters to meet a target loss minimization objective. The optimizer that is employed has a greater impact on a model's training effectiveness and efficiency [32]. DL optimizers such as Adamax, Adam, RMSProp, and Stochastic Gradient Descent are used in this study.

Adam: Adaptive Moment Estimation is an optimization approach that uses gradient descent. When confronted with an intricate problem that encompasses numerous data points or factors, employing this method

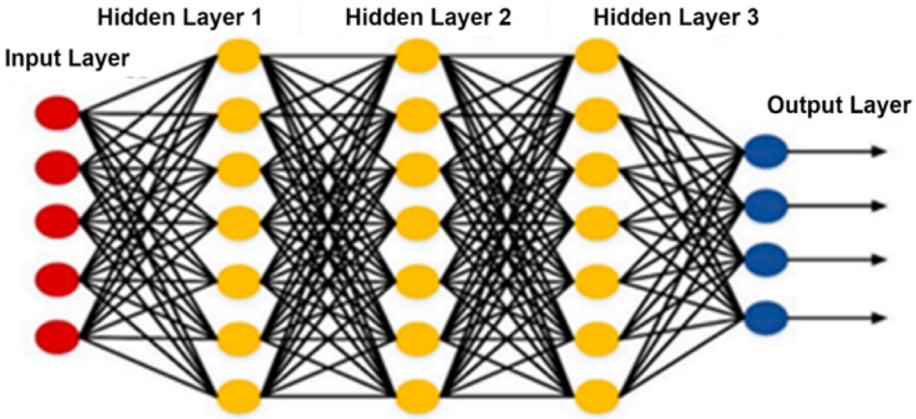

Fig. 6. DNN's structure.

demonstrates significant efficacy.

Adamax: A variant of Adam, Adam Infinity Norm uses a gradient-based optimization technique of the first order. Due to its ability to adapt the learning rate based on data properties, this model performs very well in learning time-varying processes such as speech data that are subject to continually changing noise conditions [33].

Rmsprop: Root Mean Squared Propagation (RMSprop) is an optimization approach used in DL and other types of ML. It's an improved variation of the traditional gradient descent approach, and it helps train models more quickly and reliably.

Stochastic Gradient Descent**:** When updating parameters in stochastic gradient descent (SGD), a small, randomly selected subset of data (i.e., a "mini-batch") is used instead of the full dataset [34].

Hidden Layers: Since it is between the output and input layers, this layer's nodes are hidden. They function as a surrogate for the neural network or abstraction. The input layer processes any recently added features before transmitting them to the hidden layer and then to the output layer [35]. The research employs a system model that utilizes varying numbers of hidden layers to achieve optimal output. This model utilizes either 3, 4, or 5 hidden layers.

Activation Functions: When selecting whether or not to activate a neuron, the activation function considers both the weighted sum and a bias. The linearity of a neuron's output is altered when the activation function is applied. The following variety of activation functions are used in the study [36].

Sigmoid: Provide real values as input and obtain an integer within the range of zero to one with this function. When the input is large, the resultant value should approach 1, and when the input is tiny and negative, it should approach 0. It is commonly used in scenarios involving modeling when there is a need for probability prediction.

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

Any probability can take on values between zero and one, making the sigmoid distribution a good fit. Since the function's gradient is continuous, the estimated value will not swing wildly. The sigmoid activation function is depicted visually as the letter S [37].

Relu: Despite its first appearance as a basic linear function, ReLU possesses a derivative, facilitates backpropagation, and demands minimal CPU resources. The main issue is that the ReLU function does not excite all neurons simultaneously. Neurons become inactive when the linear transformation produces a negative outcome. The following equation is the source citation of [38].

$$f(X) = \max\ (0, x) \tag{2}$$

Elu: The Exponential Linear Unit (ELU) is a form of ReLU that adjusts the function's inverse slope. In contrast to the straight lines employed by leaky ReLU and parametric ReLU, ELU employs a log curve when characterizing negative values.

$$f(x) = \begin{cases} x & for\ x \geq 0 \\ \alpha(e^x - 1) & for\ x < 0 \end{cases} \tag{3}$$

The ELU function exhibits a gradual smoothing effect until it reaches the value of - α, while the RELU function undergoes a more abrupt smoothing process [39].

Tanh: The output range of a Tanh function is −1 to 1. According to Ref. [40], as the input (a positive number) approaches −1.0 in the Tanh function, the output (a negative number) approaches 1.0, and vice versa.

$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \tag{4}$$

3.3. Machine Learning models

ML operates under the assumption that machines may be trained to autonomously acquire knowledge and enhance their performance. It is the field of research focused on developing machines that can acquire knowledge and improve their performance via experience rather than relying solely on explicit programming. This represents a significant deviation from the established conventions of computer programming [41]. In this article, the credit card default prediction system model is constructed using a DT and the Adaboost classifier.

3.3.1. Decision tree classifier

A decision tree (DT) is a hierarchical model or graph that is utilized to classify decisions and their corresponding outcomes. Every decision node in a decision tree represents an attribute test, and each branch represents a potential outcome. Every terminal node is assigned a class designation [59,60]. The peak of a tree is its root node. One can employ a DT generated from a set of training data to assign labels to training instances and extract patterns. The classification of a tuple X is established by comparing its attribute values with those contained in the DT. A path from the root leads to a leaf node, which stores the anticipated class for a given tuple. DT can be used to build classification rules with minimal complexity [42]. The following figure is the source citation of [43]. Figure 7 represents the structure of DT.

The decision-making process and classification tasks both make considerable use of the DT algorithm. Setting any parameters is not necessary while creating a DT. It functions well with data that has multiple dimensions. The most straightforward way to communicate information to people is through a tree structure. The training and sorting portions of DT are rapid and effortless. According to Refs. [43,44], this classification method demonstrates a high success rate in accurately

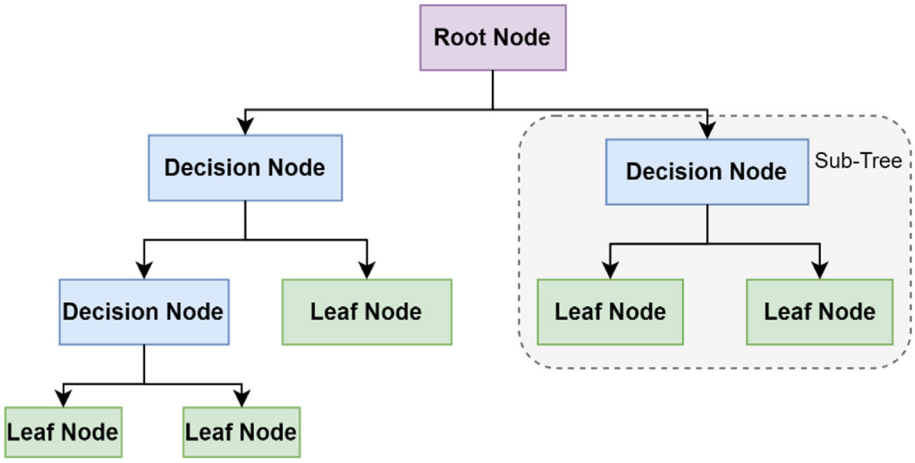

Fig. 7. Structure of DT

categorising data, and it is capable of processing both numeric and classified information.

3.3.1.1. Hyper tuning parameters. Criterion: This statistic can be used to evaluate the quality of a split in a decision tree. To calculate Gini Impurity or information gain, insert either "Gini" or "entropy". Min_samples_split: It indicates the minimum amount of samples that a node needs to split. The max_depth option can be used to set the tree's maximum depth [44].

3.3.2. Adaboost classifier

The algorithm is based on an ensemble approach that imitates boosting, in which a voting mechanism combines several weak learners to produce a stronger algorithm. Boosting involves training a series of underperforming learners sequentially on progressively modified versions of the data. As a result of employing a weak learner (G1(x)) in the training process, the initial boosting cycle may produce prediction outputs that inaccurately classify certain cases. To enhance the classification accuracy of the second weak learner, the subsequent boosting cycle assigns greater importance to misclassified data compared to correctly classified ones [45]. The mislabeled data from the second, less proficient learner has been reclassified. One of the hazards involves the potential reclassification of findings that were previously labelled. After repeating this approach M times, we merge unsuccessful learners with strong meta-learners (G(x)). The final meta-learner uses the weighted majority voting process outlined by the Equation to assign a prediction label to each record.

$$G(x) = sign\left(\sum_{m=1}^{M} \alpha_m \, G_m(x)\right) \tag{5}$$

In the above equation, α represents the significance of weak learners in the ultimate voting approach [46]. AdaBoost requires an iterative approach to solve the problem. The process commences with a period of random sampling. In the second stage, the basic classifier is trained using the subsets that have been gathered. The next step involves creating a strong classifier by combining the weighted outputs of these fundamental classifiers [47]. The following figure is the source citation of [48]. The flow of the Adaboost algorithm is shown in Figure 8.

Several weak learners, or "stumps," are fortified by AdaBoost via a sequential training procedure. These incompetent learners construct a meta-learner that relies on weighted majority voting to generate predictions. In successive boosting rounds, more weight is given to the samples that were mistakenly predicted [49].

3.3.2.1. Hypertuning parameters. **Algorithm:** Both SAMME and SAMME.R are viable choices. The performance of both SAMME and SAMME.R is assessed, and their outcomes are compared. SAMME.R incorporates probability estimations for updating the additive model, whereas SAMME solely relies on classifications [50].

n_estimators: The sample size that should be used for weak learners, which is also called base estimators. A number of 50 is used by default for n_estimator.

Learning_rate: One way to lower the input of each classifier is to change this parameter. It starts with a value of 1 [51].

4. Evaluation metrics

The effectiveness of DL and ML models is assessed through the use of evaluation measures [57]. The following evaluation metrics are considered in our experiment.

4.1. Confusion matrix

The effectiveness of a model on test data is represented by a confusion matrix [52]. The following are definitions for certain terminology frequently used in confusion matrices:

True positive (TP): The experiment yielded the expected outcome, which was consistent with the result obtained.

False positive (FP): The expected positive observational finding turned out to be negative.

True negative (TN): The observed outcome aligns with the anticipated prediction of a negative result.

False negative (FN): The observation's finding goes against the predicted negative outcome [53].

4.2. Accuracy, recall, precision, F1-Score

Precision: It is commonly known as the positive predictive value, which is the proportion of correct anticipates that were actually true out of the total number of correct predictions. Precision is the measure of correctly identified positive values expressed as a percentage [54,55].

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

Recall: The True Positive Rate (TPR) is the ratio of correctly detected True Positives to the total number [56].

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

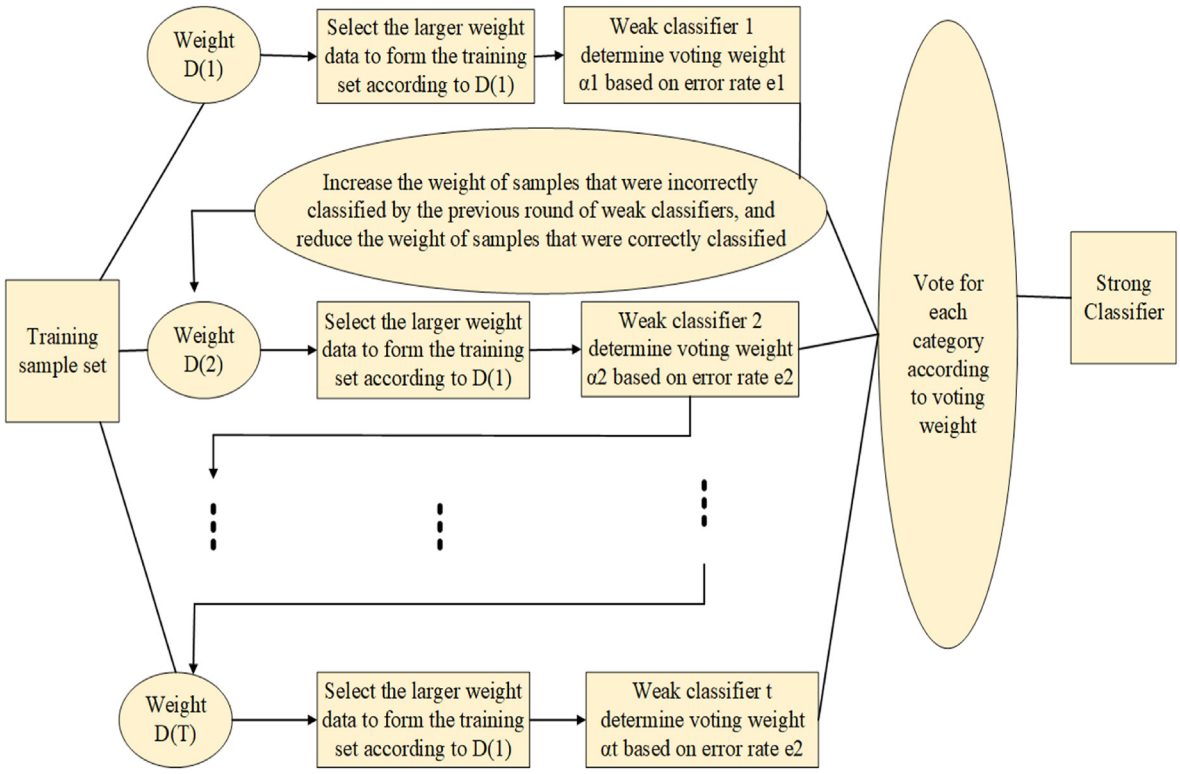

Fig. 8. Flow of AdaBoost algorithm.

F1- Score: the ideal ratio of sensitivity to accuracy.

$$F1 - Score = 2^{*}\ \frac{Precision^{*}Recall}{Precision + Reall} \tag{8}$$

Accuracy: It provides the percentage of all accurate predictions [61]:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

4.3. Deep Learning for predicting credit card default

The DL model accurately predicts a borrower's probability of credit

```
crdt_dta_MDL.summary()

Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_42 (Dense)            (None, 64)                1536

 dense_43 (Dense)            (None, 32)                2080

 dense_44 (Dense)            (None, 32)                1056

 dense_45 (Dense)            (None, 16)                528

 dense_46 (Dense)            (None, 16)                272

 dense_47 (Dense)            (None, 1)                 17

=================================================================
Total params: 5,489
Trainable params: 5,489
Non-trainable params: 0
_________________________________________________________________
```

Fig. 9. Ann layers.

Table 2
Hyperparameters and their settings.

Parameters	Activation	Loss	Optimizer	Hidden layers
Settings	relu	binary_crossentropy	adamax	5

Table 3
Results of ANN.

ANN Algorithm	Class	Precision	Recall	F1-Score	Accuracy
Performance during validation	0	0.78	0.96	0.86	75 %
	1	0.14	0.02	0.04	
Performance during testing	0	0.77	0.96	0.86	75 %
	1	0.16	0.03	0.05	

card default based on information such as the borrower's credit history, income, and expenses. Credit card issuing decisions, risk assessment, and default loss mitigation could all benefit from this model.

4.3.1. DL's performance without hypertuning

Using the training data as a guide, the sequential Keras model with six hidden layers—each with 64, 32, 32, 16, and 1 units—is used to predict credit card default. In total, 32 batches of 20 epochs are used to train the model. Figure 9 represents the ANN layers.

ReLU is the activation function for every layer. The optimizer in the following example is Adamax, and the loss function is binary cross entropy. The hyperparameters used in DL model training are displayed in the above table. Using a relu activation function, the network can use more nuanced features to make the distinction between the two groups. The 'binary_crossentropy' loss function measures the discrepancy

Table 4
ANN with varying hidden layers.

Hidden Layers	Architecture

Hidden Layers = 3	`crdt_dta_MDL.summary()`
	`Model: "sequential_8"`
	`Layer (type) Output Shape Param #`
	`dense_39 (Dense) (None, 64) 1536`
	`dense_40 (Dense) (None, 32) 2080`
	`dense_41 (Dense) (None, 32) 1056`
	`dense_42 (Dense) (None, 1) 33`
	`Total params: 4,705`
	`Trainable params: 4,705`
	`Non-trainable params: 0`

Hidden Layers = 4	`crdt_dta_MDL.summary()`
	`Model: "sequential_12"`
	`Layer (type) Output Shape Param #`
	`dense_58 (Dense) (None, 32) 768`
	`dense_59 (Dense) (None, 32) 1056`
	`dense_60 (Dense) (None, 32) 1056`
	`dense_61 (Dense) (None, 64) 2112`
	`dense_62 (Dense) (None, 1) 65`
	`Total params: 5,057`
	`Trainable params: 5,057`
	`Non-trainable params: 0`

Hidden Layers = 5	`crdt_dta_MDL.summary()`
	`Model: "sequential_13"`
	`Layer (type) Output Shape Param #`
	`dense_63 (Dense) (None, 64) 1536`
	`dense_64 (Dense) (None, 32) 2080`
	`dense_65 (Dense) (None, 32) 1056`
	`dense_66 (Dense) (None, 16) 528`
	`dense_67 (Dense) (None, 16) 272`
	`dense_68 (Dense) (None, 1) 17`
	`Total params: 5,489`
	`Trainable params: 5,489`
	`Non-trainable params: 0`

between the observed and expected values for each data point. The 'adamax' optimizer is utilized, and it employs adaptive moment estimation, a mathematical approach that quickens a network's learning as the algorithm's iteration count increases. The model's five hidden layers enable it to recognize intricate data patterns and capture more complex information. The hyperparameters and their settings can be seen in Table 2.

The model produces an output in the form of a probability, ranging from 0 to 1. The classification report presents the F1-score, precision, and recall for each class in the specified format. Table 3 shows the results of ANN.

The preceding findings concern the efficacy of an ANN model. A validation set and a testing set are used to check the accuracy of this model. As can be seen from the results, the ANN model achieves an impressive 75 % accuracy on both the validation and testing datasets. Despite this, the non-default class's (class 0) precision and recall are noticeably higher than the default class's (class 1) values. This suggests that the model has a low success rate in predicting defaults but a high success rate in predicting in predicting non-defaults.

4.3.2. DL's performance with hypertuning

Below, optimizers, hidden layers, and activation functions are adjusted to hypertune the DL model.

4.3.3. Hypertuning the model using varying hidden layers

We first use a range of hidden layers to do hyperparameter tuning on the ANN model in order to discover the optimal number of hidden layers for the DL model. The equivalent architectures for the three, four, and five hidden layers that were chosen are shown in the table below. Table 4 represents the ANN with varying hidden layers.

Both the number of parameters and hidden layers vary across models. The first model has 64 neurons in the first hidden layer, 32 in the second, and 32 in the third. This model contains 4705 parameters. There are a total of 64 neurons spread over four hidden layers in the second model. There are 5057 parameters in this model. The third model contains 5 hidden layers with the following numbers of neurons: 64, 32, 32, 16, and 16. This model contains 5489 parameters. Table 5 shows the ANN's results with varying hidden layers.

Five hidden layers make up the most accurate model (78 % accuracy), as determined by the research. The NN achieves 48 % accuracy with 3 hidden layers. The neural network isn't strong enough to adapt to new information. The accuracy, however, improved to 76 % when 4 hidden layers were used. This makes it evident that the neural network can identify more patterns in the data and, as a result, generate predictions that are more trustworthy. Accuracy increased to 78 % with 5 hidden layers, showing that the network may make even greater use of these patterns. Nevertheless, as the number of hidden layers increases, recall and accuracy for the first class (1) decline. This indicates that the model is becoming more accurate at predicting occurrences that will not result in defaults while becoming less accurate at predicting the opposite.

Table 5
ANN's results with varying hidden layers.

ANNAlgorithm	Class	Precision	Recall	F1-Score	Accuracy
Hidden layer =3	0	0.75	0.50	0.60	48 %
	1	0.18	0.40	0.25	
Hidden layer = 4	0	0.78	0.97	0.86	76 %
	1	0.11	0.01	0.02	
Hidden layer = 5	0	0.78	0.99	0.87	78 %
	1	0.26	0.01	0.02	

Table 6
ANN's result with varying optimizers.

ANN Algorithm	Class	Precision	Recall	F1-Score	Accuracy
Adam	0	0.78	1.00	0.88	78 %
	1	0.00	0.00	0.00	
RMSProp	0	0.78	0.99	0.87	77 %
	1	0.21	0.01	0.01	
Adamax	0	0.75	0.56	0.64	51 %
	1	0.19	0.36	0.25	
SGD	0	0.78	1.00	0.88	78 %
	1	0.00	0.00	0.00	

4.3.4. The model hypertuning using varying optimizers

To better predict credit card default, an ANN model is hypertuned with various optimizers including RMSProp, Adam, Adamax, and SGD. The below table is comparing the ANN model's efficiency under different optimizer parameters. The ANN results with varying optimizers can be seen in Table 6.

Adam and SGD emerged as the top-performing optimizers, attaining an accuracy rate of 78 %. The non-default class (0) likewise achieves the best levels of accuracy and recall. RMSProp, with a precision of 77 %, ranks as the second most effective optimizer. Contrary to the RMSProp optimizer, Adam exhibits inferior recall and precision for the default class. Adamax is the optimizer with the lowest level of accuracy, namely at 51 %. Additionally, it had the poorest memory and accuracy compared to both groups. Therefore, both Adam and SGD optimizers are considered leading alternatives due to their high efficiency and effectiveness in predicting credit card defaults.

4.3.5. The model hypertuning using varying activation functions

Hypertuning the ANN model with Relu, Sigmoid, Elu, and Tanh activation functions helps determine the most effective activation functions for credit card default prediction. The below table represents the results. Table 7 represents the ANN's results with varying activation functions.

According to the results presented above, the Relu activation function is only 77 % as accurate as the Sigmoid activation function. Class 1 predictions, however, could only be relied upon in 18 % of default scenarios (recall = 0.18) using the model. This is a significant improvement over the Sigmoid activation function. The Elu and Tanh activation functions, respectively, achieve accuracies of 77 % and 78 %. Class 1 recall, on the other hand, is low, at 0.09 and 0. Since the Relu activation function is superior at predicting credit card defaults, these other activation functions are not as helpful.

4.4. Using Adaboost ML algorithm for predicting credit card default

AdaBoost is a commonly employed ML technique for classifying credit card defaults. The process involves the amalgamation of weak learners, or algorithms, to generate a more robust learner overall. This study involves optimising the approach by fine-tuning several aspects,

Table 7
ANN's results with varying activation functions.

ANN Algorithm	Class	Precision	Recall	F1-Score	Accuracy
Sigmoid	0	0.78	1.00	0.88	78 %
	1	0.00	0.00	0.00	
Relu	0	0.78	0.99	0.87	77 %
	1	0.18	0.01	0.02	
Elu	0	0.78	0.99	0.87	77 %
	1	0.09	0.00	0.01	
Tanh	0	0.78	1.00	0.88	78 %
	1	0.00	0.00	0.00	

Table 8
Different hypertuned values.

Parameters	Values	Chosen parameters
algorithm	SAMME.R, SAMME	SAMME
learning_rate	1.0, 1.1, 1.2, 1.3	1.0
n_estimators	50, 60, 70, 100	50

such as the algorithm, learning rate, number of estimators, and number of tasks. The precise modifications for each parameter are detailed in the subsequent paragraphs. Table 8 represents different hyper-tuned values.

By assessing the model's resultant output over an extensive range of parameter values, Grid Search can be used to find the best Adaboost settings. The decision to go with the algorithm SAMME was made because of the widespread perception that it is more reliable than SAMME.R. Weak learners are capped at 50 to strike a middle ground between precision and complexity. When the learning rate is set to 1, both overfitting and underfitting are reduced to a minimum. The outputs of training and validating the algorithm using the aforementioned settings are tabulated below. The Adaboost performance is shown in Table 9.

Results from the testing and validation stages indicate that class 0 (no default) was tested and validated with a precision of 0.84 and 0.83, respectively. This shows that 83 % and 84 % of the instances with no default might have been correctly predicted. Class 1 (default) demonstrated a precision of 0.69 and 0.70 during the validation and testing phases, respectively. This suggests that 69 % and 70 % of default occurrences were correctly anticipated by the model. The model accurately predicts 32 % and 34 % of the default instances during the validation and testing stages, respectively, as indicated by the class 1 recalls of 0.32 and 0.34. The findings point to the AdaBoost algorithm as a reliable method for predicting credit card default. Class 1, which is the default, shows a noteworthy recall rate and a high degree of accuracy.

4.4.1. Using decision tree ML algorithm for predicting credit card default

DTs can be used to forecast the likelihood of credit card defaults. Using a DT structure, it categorises those who have defaulted on their credit cards. The decisions are based on the customer's age, income level, and credit history. In this study, we hyper-tune the algorithm by experimenting with different settings for its maximum depth, learning rate, number of tasks, and number of estimators. Table 10 represents the DL's parameters.

Decision Tree ML model parameters can be optimised with the use of a technique called Grid Search [58]. This is done by testing the model's responsiveness to varying inputs. The Gini criterion is chosen because it is often considered to be the most reliable. The number 5 was chosen as the baat after careful consideration of the trade-offs between precision and complexity. After weighing a number of options, we settled on the fewest samples required to split a node as the best compromise between accuracy and complexity. For the best compromise between overfitting and underfitting, a maximum tree depth of 5 is recommended. Table 11 shows the performance of DT.

Results from both the validation and testing phases show that the DT approach is 82 % accurate, as shown in the table. With a validation precision of 0.83 and a testing precision of 0.83, respectively, class 0 (no default) is correctly predicted in 83 % of validation cases and 83 % of test cases. Both the validation and testing precision for Class 1 (default) are 0.69, therefore approximately 69 % of default scenarios can be predicted with high confidence. Class 1 recalls of 0.32 and 0.34 show that 32 % and 34 % of the default instances are correctly expected by the model during validation and testing, respectively. Using the DT algorithm to forecast credit card default appears to be a practical strategy, as evidenced by the research results. Class 1 (the default) has a high rate of accuracy and recall in the outcomes.

Table 9
AdaBoost's performance.

Adaboost Algorithm	Class	Precision	Recall	F1-Score	Accuracy
Performance during validation	0	0.83	0.96	0.89	0.82
	1	0.69	0.32	0.43	
Performance during testing	0	0.84	0.96	0.89	0.82
	1	0.70	0.34	0.46	

Table 10
DL's parameters.

Parameters	Values	Chosen parameters
criterion	gini, entropy, log_loss	gini
max_depth	5, 15, 25	5
min_samples_split	5, 7, 9, 10	5

Table 11
Performance of DT.

Algorithm	Class	Precision	Recall	F1-Score	Accuracy
DT during validation	0	0.83	0.96	0.89	0.82
	1	0.69	0.32	0.44	
DT during testing	0	0.83	0.96	0.89	0.82
	1	0.69	0.34	0.45	

4.5. Overall experimental results

Implementing the most accurate parameter settings for ML Algorithms like DT, AdaBoost, and ANN models improves their ability to predict credit card defaults. An in-depth analysis of the DL model's performance is provided below. Table 12 shows the overall comparison of the DL.

Figure 10 shows the overall comparison of the DL models. As can be seen from the above table and figure, a DL model with five hidden layers, either the Adam or SGD optimizer, and either the sigmoid or Tanh activation function produced the maximum accuracy of 78 % when it came to credit card default prediction. When compared to models with less than four hidden layers, models trained with the Adamax or RMSProp optimizers, and models triggered with the Elu or Relu functions, this model performed better. The model may learn complex relationships between input and output variables, which can further improve the impressive capabilities of ANNs. Moreover, the utilisation of the Adam or SGD optimizers in the model enables faster convergence and higher precision, surpassing the effectiveness of the RMSProp or Adamax optimizers, which are comparatively less up-to-date and efficient. By utilising the sigmoid or tanh activation function, the model can capture a greater number of non-linear relationships, resulting in superior performance compared to the Elu and Relu activation functions in forecasting credit card defaults. Furthermore, the hyperparameters of DL were hypertuned, resulting in a direct rise in accuracy from 75 % to 78 % due to the corresponding increase in precision. The graph below displays the highest levels of accuracy attained by the ML and DL models in predicting credit card defaults.

Figure 11 represents the models' evaluation. According to the table and figure provided, the ANN model achieved an accuracy of 78 % on both the validation and testing sets. In comparison, the DT and AdaBoost ML models each achieved an accuracy of 82 %. The DT and AdaBoost ML models demonstrated superior accuracy compared to the ANN model, owing to their intrinsic simplicity and clarity. As a result, they are more

Table 12
Overall comparison of DL.

DL	Settings	Accuracy
Model without hypertuning	Default	75 %
Model variation in hidden layers	Hidden layer = 5	78 %
	Hidden layer = 4	76 %
	Hidden layer = 3	48 %
Model variation in optimizers	RMSProp	77 %
	Adam	78 %
	SGD	78 %
	Adamax	51 %
Model variation in the activation function	Relu	77 %
	Sigmoid	78 %
	Tanh	78 %
	Elu	77 %

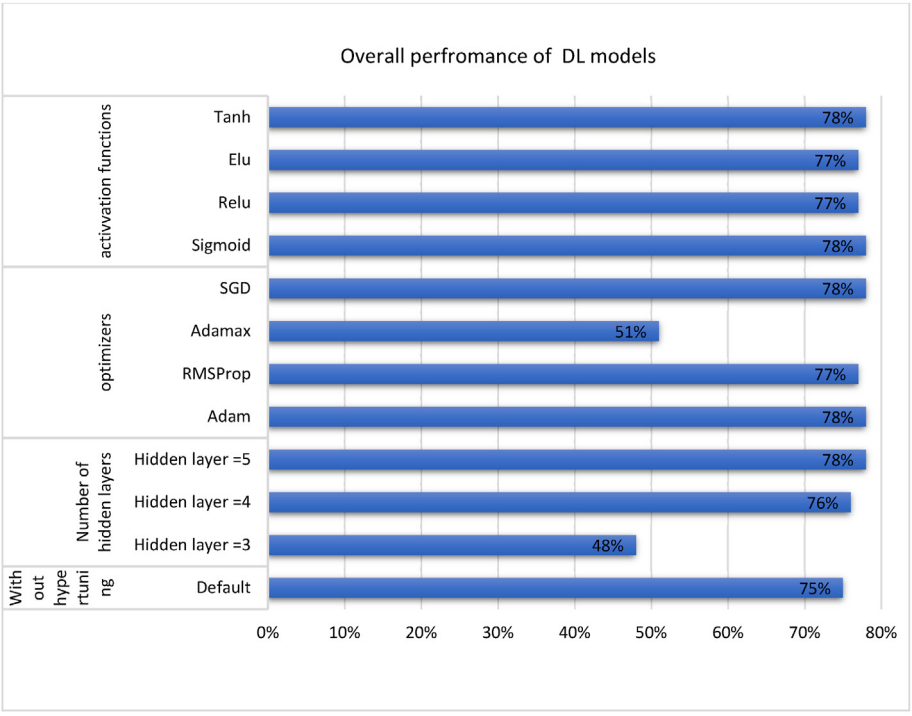

Fig. 10. Overall comparison of DL models.

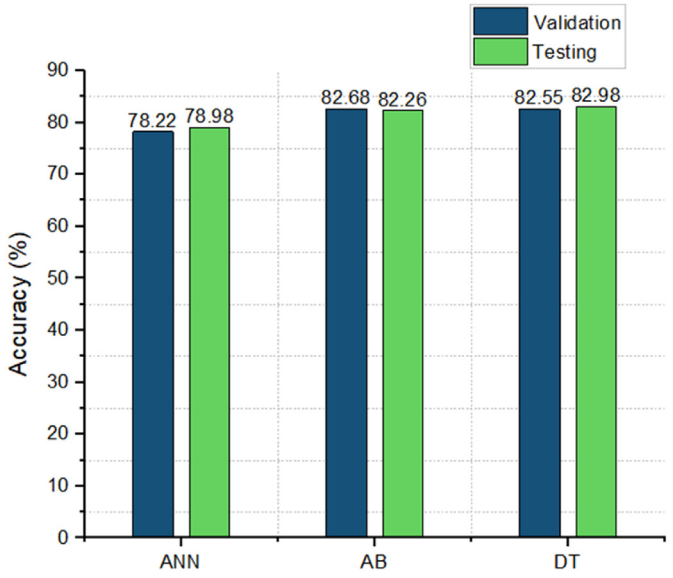

Fig. 11. Models evaluation.

efficient in illuminating the factors behind credit card defaults and are less prone to overfitting. The ANN model exhibits greater complexity and is capable of yielding very accurate outcomes. However, it also has a heightened risk of overfitting. Furthermore, in contrast to the AdaBoost and DT ML methods, the ANN model lacks the same level of interpretability.

4.6. Comparison with other research work

We summarise the results of a comparison between the conclusions of this study and those of several other researchers' more recent studies on the same topic. Comparison with other literature is shown in Table 13.

Table 13
Comparison with other literature.

Author	Model	Proposed objectives	Results
[15] Thennakoon et al., 2019	NB, LR, and SVM	This research aims to uncover four notable cases of fraud that occurred during actual transactions.	The SVM model achieves a peak accuracy of 91 % in detecting and classifying 4 instances of fraudulent scenarios.
[17] Yee et al., 2018	NB logistics and J48 classifiers, K2, Tree Augmented Naive Bayes	A study has been proposed to explore the application of data mining and ML in order to prevent losses caused by fraudulent activity.	Every Bayesian classifier achieved an accuracy of over 95 %.
[19] Lakshmi et al., 2018	DT, RF, and LR	The efficiency of three distinct strategies for identifying fraudulent credit card activity is compared in this study.	The DT, LR, and RF classifiers achieved accuracies of 94.3 %, 90.0 %, and 95.5 % respectively.
[20] Trivedi et al., 2020	KNN, SVM, ANN, RF, LR, NB, and GBM	This study offers a trustworthy method for identifying credit card theft that makes use of ML and feedback systems.	The accuracy percentage of RF approaches is 95.988 %.

4.7. Study limitation

The dataset utilized in this research may not be representative of the complete spectrum of credit card default forecasts because of its modest size. Limiting the analysis to only three DL characteristics means that not all possible factors that influence the precision of credit card default prediction have been taken into account. The scope of this research is restricted to the analysis of ANN, DT, and Adaboost algorithms. Results

from different algorithms may be more reliable. The sample size of the dataset is so limited that any inferences drawn from it may be somehow biased.

5. Conclusion and future work

Credit card payments are accepted for online transactions since the method is safe and user-friendly. With the surge in popularity of credit card use comes an increase in credit card fraud. Many sums are lost by victims and banks alike when credit card fraud happens. Accurately identifying fraudulent transactions in real-time is one of a credit card fraud detection system's most crucial feature. As a result of developments in AI tools, ML and DL algorithms are being used to anticipate credit card theft. Therefore, this study focuses on the use of the DL model such as ANN models ML models like AdaBoost and the DT approach to detect consumers who have fallen behind on their credit card payments. The default of credit card client's dataset was obtained from the UCI ML library to run the experiment. The next step is to apply various EDA methods to the raw data to prepare it for visualization. In addition, we use ML models such as AdaBoost and DT ML methods in addition to an ANN model with adjustable hyperparameters to partition the data and analyze it. This research demonstrates that the DT and AdaBoost ML models outperformed the ANN model. The ANN model, consisting of 5 hidden layers, was trained using either the Adam or SGD optimizer. The activation function used was either sigmoid or Tanh. This configuration resulted in the highest accuracy of 78 %. Furthermore, the implementation of an AdaBoost resulted in an accuracy rate of 82 % for both the validation and testing datasets. The DT attained an accuracy rate of 82 % on both the validation and testing datasets. This exemplifies the efficacy of employing ML algorithms to identify instances of credit card default.

Future work on this investigation will analyze the model's hyperparameters in an effort to achieve perfect precision, accuracy, and recall. Developing DL models that can detect credit cards in real-time, across different media types including images and video, is the next expected step for this research. This would be advantageous for applications such as fraud detection and safety surveillance. The detection of credit cards in unorganized data like PDFs and emails is also a major focus of this study on DL models. Online transactions and application processing would both benefit from this. We also intend to use Explainable Artificial Intelligence (XIA) techniques for the interpretability of the models in the future work.

CRediT authorship contribution statement

Fazal Wahab: Conceptualization, Data curation, Formal analysis, Methodology, Validation, Writing – original draft. **Imran Khan:** Formal analysis, Supervision, Validation, Writing – review & editing. **Sneha Sabada:** Conceptualization, Data curation, Investigation, Resources, Software, Visualization.

Declaration of competing interest

The authors do not have any financial or proprietary interests in the material mentioned in this article.

Acknowledgment

No grants, funds, or other support was received.

References

[1] T. Sun, M.A. Vasarhelyi, Predicting credit card delinquencies: an application of deep neural networks, Intell. Syst. Account. Finance Manag. 25 (4) (2018) 174–189.
[2] Y. Sayjadah, I.A.T. Hashem, F. Alotaibi, K.A. Kasmiran, Credit card default prediction using machine learning techniques, in: 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA), IEEE, October 2018, pp. 1–4.
[3] H.W. Teng, M. Lee, Estimation procedures of using five alternative machine learning methods for predicting credit card default, Rev. Pac. Basin Financ. Mark. Policies 22 (3) (2019) 1950021.
[4] N. Syam, A. Sharma, Waiting for a sales renaissance in the fourth industrial revolution: machine learning and artificial intelligence in sales research and practice, Ind. Market. Manag. 69 (2018) 135–146.
[5] S.A. Ebiaredoh-Mienye, E. Esenogho, T.G. Swart, Artificial neural network technique for improving prediction of credit card default: a stacked sparse autoencoder approach, Int. J. Electr. Comput. Eng. 11 (5) (2021) 4392.
[6] L. Gui, Application of Machine Learning Algorithms in Predicting Credit Card Default Payment, University of California, Los Angeles, 2019. Master's thesis.
[7] T.M. Alam, K. Shaukat, I.A. Hameed, S. Luo, M.U. Sarwar, S. Shabbir, J. Li, M. Khushi, An investigation of credit card default prediction in the imbalanced datasets, IEEE Access 8 (2020) 201173–201198.
[8] D. Tanouz, R.R. Subramanian, D. Eswar, G.P. Reddy, A.R. Kumar, C.V. Praneeth, Credit card fraud detection using machine learning, in: 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), IEEE, May 2021, pp. 967–972.
[9] O. Adepoju, J. Wosowei, H. Jaiman, Comparative evaluation of credit card fraud detection using machine learning techniques, in: 2019 Global Conference for Advancement in Technology (GCAT), IEEE, 2019, pp. 1–6.
[10] N.S. Alfaiz, S.M. Fati, Enhanced credit card fraud detection model using machine learning, Electronics 11 (4) (2022) 662.
[11] Y. Chen, R. Zhang, Research on credit card default prediction based on k-means SMOTE and BP neural network, Complexity (2021) 1–13.
[12] J. Gao, W. Sun, X. Sui, Research on default prediction for credit card users based on XGBoost-LSTM model, Discrete Dynam Nat. Soc. (2021) 1–13.
[13] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, A. Anderla, Credit card fraud detection-machine learning methods, in: 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH), IEEE, March 2019, pp. 1–5.
[14] V.N. Dornadula, S. Geetha, Credit card fraud detection using machine learning algorithms, Procedia Comput. Sci. 165 (2019) 631–641.
[15] A. Thennakoon, C. Bhagyani, S. Premadasa, S. Mihiranga, N. Kuruwitaarachchi, Real-time credit card fraud detection using machine learning, in: 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, January 2019, pp. 488–493.
[16] S.P. Maniraj, A. Saini, S. Ahmed, S. Sarkar, Credit card fraud detection using machine learning and data science, Int. J. Eng. Res. 8 (9) (2019) 110–115.
[17] O.S. Yee, S. Sagadevan, N.H.A.H. Malim, Credit card fraud detection using machine learning as data mining technique, J. Telecommun. Electron. Comput. Eng. 10 (1–4) (2018) 23–27.
[18] R. Sailusha, V. Gnaneswar, R. Ramesh, G.R. Rao, Credit card fraud detection using machine learning, in: 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), IEEE, May 2020, pp. 1264–1270.
[19] S.V.S.S. Lakshmi, S.D. Kavilla, Machine learning for credit card fraud detection system, Int. J. Appl. Eng. Res. 13 (24) (2018) 16819–16824.
[20] N.K. Trivedi, S. Simaiya, U.K. Lilhore, S.K. Sharma, An efficient credit card fraud detection model based on machine learning methods, International Journal of Advanced Science and Technology 29 (5) (2020) 3414–3424.
[21] A.S. Lundervold, A. Lundervold, An overview of deep learning in medical imaging focusing on MRI, Z. Med. Phys. 29 (2) (2019) 102–127.
[22] D. Choi, C.J. Shallue, Z. Nado, J. Lee, C.J. Maddison, G.E. Dahl, On empirical comparisons of optimizers for deep learning. arXiv Preprint arXiv:1910.05446, 2019.
[23] A. Bansal, N.K. Garg, Urban sound classification using adaboost, in: International Conference on Innovative Computing and Communications: Proceedings of ICICC, vol. 1, Springer Nature, Singapore, 2022, pp. 621–631. September 2022.
[24] A. Subasi, D.H. Dammas, R.D. Alghamdi, R.A. Makawi, E.A. Albiety, T. Brahimi, A. Sarirete, Sensor based human activity recognition using adaboost ensemble classifier, Procedia Comput. Sci. 140 (2018) 104–111.
[25] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U.R. Acharya, V. Makarenkov, A review of uncertainty quantification in deep learning: techniques, applications and challenges, Inf. Fusion 76 (2021) 243–297.
[26] G. Nguyen, S. Dlugolinsky, M. Bobák, V. Tran, Á. López García, I. Heredia, P. Malík, L. Hluchý, Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey, Artif. Intell. Rev. 52 (2019) 77–124.
[27] A. Kamilaris, F.X. Prenafeta-Boldú, Deep learning in agriculture: a survey, Comput. Electron. Agric. 147 (2018) 70–90.
[28] S. Nosratabadi, A. Mosavi, P. Duan, P. Ghamisi, F. Filip, S.S. Band, U. Reuter, J. Gama, A.H. Gandomi, Data science in economics: comprehensive review of advanced machine learning and deep learning methods, Mathematics 8 (10) (2020) 1799.
[29] N. Dimitriou, O. Arandjelović, P.D. Caie, Deep learning for whole slide image analysis: an overview, Front. Med. 6 (2019) 264.
[30] E. Perez, F. Strub, H. De Vries, V. Dumoulin, A. Courville, Film: visual reasoning with a general conditioning layer, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018, 1.
[31] P.R. Kumar, E.B.K. Manash, Deep learning: a branch of machine learning, in: Journal of Physics: Conference Series, vol. 1228, IOP Publishing, 2019 012045, 1.
[32] X. Kan, Y. Fan, Z. Fang, L. Cao, N.N. Xiong, D. Yang, X. Li, A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network, Inf. Sci. 568 (2021) 147–162.
[33] D. Yi, J. Ahn, S. Ji, An effective optimization method for machine learning based on ADAM, Appl. Sci. 10 (3) (2020) 1073.

[34] Kumar, A., Sarkar, S., and Pradhan, C. "Malaria disease detection using CNN technique with sgd, rmsprop and adam optimizers." In Deep Learning Techniques for Biomedical and Health Informatics, 211-230.
[35] P.A. Pattanaik, M. Mittal, M.Z. Khan, Unsupervised deep learning cad scheme for the detection of malaria in blood smear microscopic images, IEEE Access 8 (2020) 94936–94946.
[36] G. Currie, E. Rohren, Intelligent imaging in nuclear medicine: the principles of artificial intelligence, machine learning and deep learning, Semin. Nucl. Med. 51 (2) (2021) 102–111.
[37] S. Langer, Approximating smooth functions by deep neural networks with a sigmoid activation function, J. Multivariate Anal. 182 (2021) 104696.
[38] G. Lin, W. Shen, Research on convolutional neural network based on improved Relu piecewise activation function, Procedia Comput. Sci. 131 (2018) 977–984.
[39] M. Cococcioni, F. Rossi, E. Ruffaldi, S. Saponara, A novel posit-based fast approximation of elu activation function for deep neural networks, in: 2020 IEEE International Conference on Smart Computing (SMARTCOMP), IEEE, September 2020, pp. 244–246.
[40] S.L. Shen, N. Zhang, A. Zhou, Z.Y. Yin, Enhancement of neural networks with an alternative activation function tanhLU, Expert Syst. Appl. 199 (2021) 117181.
[41] N. Rawindaran, A. Jayal, E. Prakash, C. Hewage, Cost benefits of using machine learning features in NIDS for cyber security in UK small medium enterprises (SME), Future Internet 13 (8) (2021) 186.
[42] C.L. Lin, C.L. Fan, Evaluation of CART, CHAID, and QUEST algorithms: a case study of construction defects in Taiwan, J. Asian Architect. Build Eng. 18 (6) (2019) 539–553.
[43] B. Charbuty, A. Abdulazeez, Classification based on decision tree algorithm for machine learning, Journal of Applied Science and Technology Trends 2 (1) (2021) 20–28.
[44] X.W. Liu, Z.L. Long, W. Zhang, L.M. Yang, Key feature space for predicting the glass-forming ability of amorphous alloys revealed by a gradient boosted decision trees model, J. Alloys Compd. 901 (2022) 163606.
[45] S.J. Lee, T. Chen, L. Yu, C.H. Lai, Image classification based on the boost convolutional neural network, IEEE Access 6 (2018) 12755–12768.
[46] O. Sagi, L. Rokach, Ensemble learning: a survey, Wiley Interdisciplinary Reviews: Data Min. Knowl. Discov. 8 (4) (2018) e1249.
[47] A. Ibrahim, A. Tharwat, T. Gaber, A.E. Hassanien, Optimized superpixel and AdaBoost classifier for human thermal face recognition, Signal, Image and Video Processing 12 (2018) 711–719.
[48] S. Wei, L. Zhu, L. Chen, Q. Lin, An adaboost-based intelligent driving algorithm for heavy-haul trains, Actuators 10 (8) (2021) 188.
[49] R.S. Khairy, A.S. Hussein, H.T. Salim Alrikabi, The detection of counterfeit banknotes using ensemble learning techniques of AdaBoost and voting, International Journal of Intelligent Engineering & Systems 14 (1) (2021).
[50] Y. Zhou, T.A. Mazzuchi, S. Sarkani, M-AdaBoost-A based ensemble system for network intrusion detection, Expert Syst. Appl. 162 (2020) 113864.
[51] M. Bansal, A. Goyal, A. Choudhary, A comparative analysis of K-nearest neighbor, genetic, support vector machine, decision tree, and long short-term memory algorithms in machine learning, Decision Analytics Journal 3 (2022) 100071.
[52] F. Wahab, I. Khan, T. Hussain, A. Amir, An investigation of cyber attack impact on consumers' intention to purchase online, Decision Analytics Journal (2023) 100297.
[53] A.E. Maxwell, T.A. Warner, L.A. Guillén, Accuracy assessment in convolutional neural network-based deep learning remote sensing studies—part 1: a literature review, Rem. Sens. 13 (13) (2021) 2450.
[54] F. Wahab, I. Ullah, A. Shah, et al., Design and implementation of real-time object detection system based on single-shoot detector and OpenCV, Front. Psychol. 13 (2022) 1039645.
[55] P. Tiwari, S. Mehta, N. Sakhuja, J. Kumar, A.K. Singh, Credit card fraud detection using machine learning: a study, arXiv preprint arXiv:2108.10005 (2021).
[56] X. Zhang, Y. Han, W. Xu, Q. Wang, HOBA: a novel feature engineering methodology for credit card fraud detection with a deep learning architecture, Inf. Sci. 557 (2021) 302–316.
[57] Shah, B. Ali, F. Wahab, I. Ullah, F. Alqahtani, A. Tolba, A three-way clustering mechanism to handle overlapping regions, IEEE Access 12 (2024) 6546–6559, https://doi.org/10.1109/ACCESS.2024.3349620.
[58] T.T. Nguyen, H. Tahir, M. Abdelrazek, A. Babar, Deep learning methods for credit card fraud detection, arXiv preprint arXiv:2012.03754 (2020).
[59] F. Wahab, I. Khan, K. Si, Investigating offline password attacks: a comprehensive review of rainbow table techniques and countermeasure limitations, Romanian Journal of Information Technology and Automatic Control, ISSN 1220-1758 34 (1) (2024) 81–96, https://doi.org/10.33436/v34i1y202408.
[60] I.U. Khan, A. Usoro, M. Crowe, Is organisational culture a hurdle in online knowledge sharing? A conceptual view, Int. J. Bus. Inf. Syst. (2021) 336–347.
[61] A. Shah, B. Ali, F. Wahab, et al., Entropy-based grid approach for handling outliers: a case study to environmental monitoring data, Environ. Sci. Pollut. Control Ser. (2023) 1–20.