Asian Journal of Engineering and Applied Technology
ISSN: 2249-068X (P) Vol.11 No.2, 2022, pp.1-10

DOI: https://doi.org/10.51983/ajeat-2022.11.2.3348

Utilizing Prediction Intervals for Unsupervised Detection of Fraudulent Transactions: A Case Study

Isuru Hewapathirana
Senior Lecturer, Department of Software Engineering, University of Kelaniya, Sri Lanka
E-mail: ihewapathirana@kln.ac.lk
(Received 5 May 2022; Revised 11 June 2022; Accepted 18 July 2022; Available online 2 August 2022)

Abstract - **Money laundering operations have a high negative impact on the growth of a country's national economy. As all financial sectors are increasingly being integrated, it is vital to implement effective technological measures to address these fraudulent operations. Machine learning methods are widely used to classify an incoming transaction as fraudulent or non-fraudulent by analyzing the behaviour of past transactions. Unsupervised machine learning methods do not require label information on past transactions, and a classification is made solely based on the distribution of the transaction. This research presents three unsupervised classification methods: ordinary least squares regression-based (OLS) fraud detection, random forest-based (RF) fraud detection and dropout neural network-based (DNN) fraud detection. For each method, the goal is to classify an incoming transaction amount as fraudulent or non-fraudulent. The novelty in the proposed approach is the application of prediction interval calculation for automatically validating incoming transactions. The three methods are applied to a real-world dataset of credit card transactions. The fraud labels available for the dataset are removed during the model training phase but are later used to evaluate the performance of the final predictions. The performance of the proposed methods is further compared with two other unsupervised state-of-the-art methods. Based on the experimental results, the OLS and RF methods show the best performance in predicting the correct label of a transaction, while the DNN method is the most robust method for detecting fraudulent transactions. This novel concept of calculating prediction intervals for validating an incoming transaction introduces a new direction for unsupervised fraud detection. Since fraud labels on past transactions are not required for training, the proposed methods can be applied in an online setting to different areas, such as detecting money laundering activities, telecommunication fraud and intrusion detection.**
Keywords: **Fraud Detection, Prediction Intervals, Ordinary Least Squares, Random Forest, Dropout Neural Network, Unsupervised Machine Learning**

I. INTRODUCTION

As the world becomes more interconnected and information moves more quickly, opportunities for criminals to misuse the financial system are also increasing. Fraud activities such as sanctions, insider trading, terrorist financing, bribery, corruption, and money laundering, involve criminals obtaining money, services or property, illegally or through deception, and then profiting from the proceeds. Due to its substantial increment in occurrence, credit card fraud is one type of money laundering operation that has attracted much attention from the scientific research community [1], [2]. Credit card fraud is the unauthorized use of a person's credit card to make fraudulent purchases without the user's knowledge. Traditional approaches, such as manual auditing, were employed to detect suspicious transactions [3]. However, nowadays, these methods have become inefficient and unreliable due to the high occurrence of fraud and the severity of repercussions. Most financial sector organizations have adapted sophisticated security systems to monitor customers' regular spending habits, and flag transactions that deviate noticeably from the usual. Recently, machine learning-based approaches have gained much popularity for this purpose. A few examples of utilized ML models include Support Vector Machines [4], K-Nearest Neighbours [4], Self-Organising Maps [5], Random Forests [6], Neural networks [7] and Bayesian algorithms [8]. Machine learning-based methods can be mainly divided into supervised and unsupervised methods. Supervised machine learning requires labelled transactions to train the machine learning model and classify an incoming transaction as fraudulent or non-fraudulent. Unsupervised machine learning, on the other hand, examines unlabeled transaction data and identifies outliers or transactions that deviate considerably from the rest as potentially fraudulent. Overall, both supervised and unsupervised methods can be used to classify a transaction as fraudulent or non-fraudulent.

Even though most of the research related to credit card fraud detection utilizes supervised methods, for real-world applications, labelled data is not readily available. Hence, unsupervised methods are preferable. Furthermore, an online learning approach, where the model is updated as new data arrives, is more appropriate for fraud detection. Conducting online learning on a supervised method becomes challenging as obtaining label information requires a lot of time and effort. Thus, unsupervised methods are more suitable for online learning of evolving user behaviour. Considering the latter-mentioned issues, we develop three unsupervised, online learning methods to validate an incoming transaction as fraudulent or non-fraudulent. The methods are unsupervised as the fraudulent or non-fraudulent labels are not incorporated for training the machine learning models. The proposed methods consider the transaction amount as the dependent variable and other transaction features as independent variables. By feeding recent past transaction data into the machine learning model employed in each method, user behaviour patterns are learned to build a

prediction interval for the transaction amount. The novel idea of utilizing prediction intervals for validating an incoming transaction amount as fraudulent or non-fraudulent is presented in this paper. An incoming transaction that falls outside its prediction interval is flagged as fraudulent. The overall effectiveness of the proposed methods will be primarily influenced by the confidence level defined for the interval calculation.

The remaining sections of the paper are as follows. Section II provides a brief review of the literature on fraud detection and prediction interval calculation methods. Section III describes the methodology and proposed approach. Experimental results are shown and discussed in Section IV. Finally, Section V summarizes the paper and discusses encountered issues and suggestions for future work.

II. LITERATURE REVIEW

This section provides a detailed outlook into the available literature on credit card fraud detection using machine learning methods. Since the suggested methodology for fraud detection in this paper is linked with calculating prediction intervals, this section will also detail its previous research.

A. Fraud Detection

Statistics and machine learning (ML) are popular technologies for detecting credit card fraud because it requires extracting information from large and complex datasets. In the past, traditional statistical methods such as linear discriminant analysis [9] have provided successful results for many applications. In recent years, more powerful supervised and unsupervised ML models have been widely used. In [10], two different random forest classifiers are trained to model the behviour of credit card transactions and generate alerts, then utilize these alerts as labels to update and train the classifier. They also claim precision to be the most suitable evaluation metric for credit card fraud detection. In [11], a comparison is made between six different artificial neural network (ANN) methods and seven different logistic regression (LR) methods based on the accuracy of classifying credit card transactions as fraudulent or non-fraudulent. Their results show the superiority of the ANN-based methods over the LR-based methods when the distribution of the training dataset is not biased towards non-fraudulent samples. The performance of four supervised machine learning (ML) methods, decision trees, K-Nearest Neighbors, logistic regression and neural network, are compared in [12] for credit card fraud detection. The data used are financial transactions of an e-commerce organization with 100,000 data samples and 9 features. Based on performance metrics such as accuracy, sensitivity, and specificity, they show that no one method performs better than another. Instead, they propose performance improvement using a model that combines multiple ML methods. In our research, the same metrics are used for evaluating the performance of the proposed fraud detection methods. Some evaluation metrics also focus on the overall benefit of the fraud detection algorithm rather than just the reliability of the prediction. In [13], the overall savings is compared against the cost of using no algorithm for calculating the performance measure. However, for calculating such a performance measure, additional data need to be collected on the cost of each prediction.

Several feature aggregation strategies are introduced, and different combinations of features are used for grouping transactions [13]. The von Mises probability distribution [14] is used to model the periodic behaviour of a transaction time. A new feature indicating whether a transaction time is within the acceptable confidence limit is also added. Comparisons are made for different sets of features (raw, aggregated, extended aggregated and periodic), using two kinds of classification algorithms; cost-insensitive method [15] and example-dependent cost-sensitive method [16] are made using a real credit card fraud dataset. The results show a considerable performance increase when recent user behaviour is considered when complex relations between features are considered, and when periodic behaviour of time is considered. The credit card transaction dataset used in [13] contains raw features and is utilized for experimenting with the proposed feature engineering strategies. However, real-world datasets with raw features are hardly made publicly available due to confidentiality reasons. Thus, it becomes impossible to apply and reproduce the proposed feature engineering strategies in the new research.

Some unsupervised approaches are found in the literature for real-world fraud detection systems. For example, a hidden Markov model (HMM) is trained in [17] on recent past data, and an incoming transaction not accepted by the HMM with a high probability is flagged as fraudulent. In the paper [18], fraudulent observations were detected as multivariate outliers by plotting the ordered squared robust Mahalanobis distances of the observations against the empirical distribution function. Although the performance of the robust Mahalanobis distance method showed lesser performance compared to the other supervised methods experimented with, the authors argue that the Mahalanobis method is helpful since it does not require labelled data to be trained and can detect fraudsters using the minimum covariance determinant matrix of the data. Two unsupervised methods, peer group analysis and break point analysis, are presented in [19] for monitoring longitudinal data to identify behavioural changes in users and detect fraudsters. Principal component analysis and the simple k-means algorithm is utilized in [20] for presenting a simple and transparent fraud detection system that also considers the geographic position of both transactions and clients. The Isolation Forest (IF) [21] and the Local Outlier Factor (LOF) [22] are two algorithms that have been utilized in [23] to detect fraudulent transactions in an unsupervised setting. The primary goal of these methods is to learn the underlying structure of the data and classify anomalous observations as fraudulent transactions. The IF algorithm detects global outliers, while the LOF algorithm performs well in detecting localized outliers. In [24], an ensemble learning method, combining both LOF and IF

algorithms, is proposed for anomaly detection in real-world complex datasets. The experimental results showed superior performance of the ensemble learning method compared to the single methods. In our research, the IF and LOF algorithms are compared against the proposed methods to evaluate fraud detection performance.

A simple moving average technique that considers recent past transactions is applied in [25] to detect suspicious customers and abnormal transactions. Transaction amount, transaction frequency and proportion of credit and debit amount are tracked over time using three moving window approaches called dynamic transaction moving average (DTMA), dynamic frequency moving average (DFMA) and transaction character (TC), respectively. In the DTMA approach, for example, when the difference between the moving average value and the current transaction value is more significant than the threshold value, an alert referred to as a 'blip' is generated. The threshold value is calculated based on the historical transactions of past confirmed suspicious customers. The approach is tested and verified on real customers' transaction data from a bank. In future, the described techniques can be applied in the mobile network domain to identify illegal use of phone calls. Similar to the moving window approach used for the DTMA method, our proposed methods also train the machine learning models on the features of recent past transactions and estimate the transaction amount for the current transaction. However, unlike calculating a single-point estimate in the DTMA method, we calculate an interval estimate to validate the current transaction value.

The work of [25] is extended in [26] to identify the inherent relationships between suspicious customers using social network analysis (SNA). After a set of customers is indicated as suspicious using the method in [25], semantic data of both normal and suspicious customers are utilized to capture associations and build a network graph, and then SNA techniques such as degree centrality and clustering are applied to extract information.

In this paper, the proposed methods combine machine learning modelling with prediction interval calculation for unsupervised detection of fraudulent transactions. Thus, the next section will focus on the related literature on calculating prediction intervals for the relevant machine-learning models.

B. Prediction Intervals

A prediction interval provides an estimate of the uncertainty associated with a point estimate and can be used to provide a range of values within which the variable is estimated to be. Several probabilistic and non-probabilistic approaches can be found for calculating the uncertainty associated with a model output [27]-[29]. One approach uses the statistical properties of the model errors observed during model training to estimate the uncertainty of the model in terms of prediction intervals [30]. In the current paper, we adopt this approach to estimate prediction uncertainty by calculating prediction intervals. The following definition for prediction intervals is taken from [31]:

Definition 1 (Prediction Interval): Let $\alpha \in [0,1]$. Then, a *(1-α)100%* prediction interval for the value of an observation is an interval constructed by a procedure such that (1-α)100% of the prediction intervals constructed by the procedure contain the true individual value of interest.

Prediction intervals calculated for ordinary least squares regression models can be categorized to two main types as discussed in [30]: (i) classical closed form prediction intervals [32], and (ii) bootstrap prediction intervals [33]. Although calculating the classical closed-form prediction intervals is quick and straightforward, its validity depends heavily on the assumption that the noise is normally distributed. Thus, this assumption is generalized in [34] by presenting a large sample prediction interval calculation method that does not require knowing the noise distribution of the regression model.

When using random forests to predict a quantitative response, an important but often overlooked challenge is the determination of prediction intervals [35]. Due to the complex structure of random forests, estimating the variability of the prediction is a challenging problem. Quantile regression forests (QRF) [36] estimate the distribution of the conditional quantiles, and the upper and lower quantiles of this distribution are used to construct the prediction interval for an unknown response. Here, the scale or shape of the conditional distribution does not need to be constant across feature values because the conditional response distribution is separately estimated using data locally to the point of interest in the feature space. However, as each conditional response distribution is separately estimated using a small amount of data local to a point, the resulting QRF intervals are often wide. They may not address the requirements of certain applications. Rather than considering only data local to a point, all training observations used to construct the random forest are utilized in [37] and [35] for constructing prediction intervals using out-of-bag samples.

In [37], the mean squared prediction error (MSPE) is calculated to estimate the variability of a prediction. Based on the experiments performed in [37], prediction intervals constructed using MSPE is preferable when the response distribution is not clustered but has constant variance across the feature space. In [35], the distribution of the random forest prediction error is approximated by the empirical distribution of out-of-bag prediction errors, and the quantiles of this distribution are used to calculate the prediction interval. Simulation studies and analysis of 60 real datasets are used to compare the finite-sample properties of the proposed interval with QRFs [36], and another recently proposed prediction interval calculation method called split conformal intervals [38]. The results indicate that intervals calculated using the method in [35] are narrower than those

of competing methods and still maintain marginal coverage rates approximately equal to nominal levels. Thus, out-of-bag intervals should be used alongside a random forest point prediction to provide a range of reasonable response values for those concluding data.

Dropout is a technique that is commonly used to avoid overfitting when training neural networks. Here, a random subset of neurons in a network layer is ignored with their incoming and outgoing connections at every input of a training observation. Thus, training an ANN with dropout can be thought of as approximating the training of an ensemble of networks that share the parameters (also called weights) for outgoing connections corresponding to units that survived dropout in each network [39].

Although dropout is usually used during the training phase of the network, [40] develops a new theoretical framework and shows how the dropout technique can be applied during testing to estimate the variability of a predicted response for a given test observation. The idea is to use a single ANN at test time with each trained weight multiplied by the probability used to retain the corresponding hidden unit during the training phase with dropout. We refer to this ANN as the dropout neural network. The work in [40] shows that a dropout neural network is mathematically equivalent to an approximation to the probabilistic deep Gaussian process [41] and hence can be used to quantify the uncertainty of the predictions from the network.

The test data is passed through the dropout neural network $B(\geq 1000)$ number of times to obtain B number of predictions from the model. By the central limit theorem [42], if there are more than 30 predictions for the same observation, the mean prediction is spread out in a normal distribution. Thus, the prediction distribution is analyzed to calculate a prediction interval for each prediction. This technique is formally known as MC Dropout in [40]. It can be used to deal with model uncertainty and misspecification at prediction time for any ANN architecture, without making any changes to how the network is trained.

There is hardly any research on the use of prediction intervals combined with machine learning for fraud detection in transaction datasets. In Section III, the methodology underlying the proposed methods is explained by introducing the data set used in this study (Section III.B), explaining the three unsupervised methods implemented for detecting fraudulent transactions (Section III.C) and discussing the performance measures used for evaluating the methods (Section III.D).

III. METHODOLOGY

This section provides a thorough discussion on the methodology for the unsupervised fraud detection methods proposed in this paper. This methodology is unique due to the utilization of prediction intervals for validating incoming transactions.

A. Preliminaries

In addition to using standard conventions in statistical notation, the following linear algebra notation will be used throughout this paper. A scalar is denoted by a lowercase letter, e.g., $\boldsymbol{x}$. A vector is denoted by a lowercase, boldface letter, e.g., x. A matrix is denoted by an uppercase letter, e.g., X. We assume that a dataset consist of $\boldsymbol{p}$ independent variables called features, a dependent variable also called as the response and $\boldsymbol{n}$ number of observations also called as samples. What is referred to as the training set is the portion of the dataset that is used for training the models. An incoming data point that has not been used for training the model will be referred to as a test observation or test sample.

B. Data Description

The input data consists of data on 284,807 transactions made by European cardholders in September 2013. There are 31 variables in the dataset. The variable 'Class' is a binary variable that describes the label for each transaction as '1' in case of fraud and '0' otherwise. There are only 492 fraudulent transactions which is only 0.172% of the overall transactions. The variable 'Time' describes the number of seconds that has passed between each transaction and the first transaction in the dataset. 'Amount' is the transaction amount considered as the dependent variable for the experiments. The rest of the 28 input variables consist of numerical values that are the result of a principal component analysis (PCA) transformation due to confidentiality reasons. The variable, 'Class' is not considered for training our models. However, as will be discussed in Section III.D, those labels are utilized for model evaluation. When visualizing the distribution of the 'Time' variable in Figure 1, it is apparent that fraudulent and non-fraudulent transactions have almost similar distributions. This implies that the 'Time' variable may not be helpful in distinguishing fraudulent transactions from non-fraudulent transactions. Thus, the 'Time' variable is not included in our predictive models. In summary, the 28 PCAs constitute the features for our models while the 'Amount' variable is considered as the response.

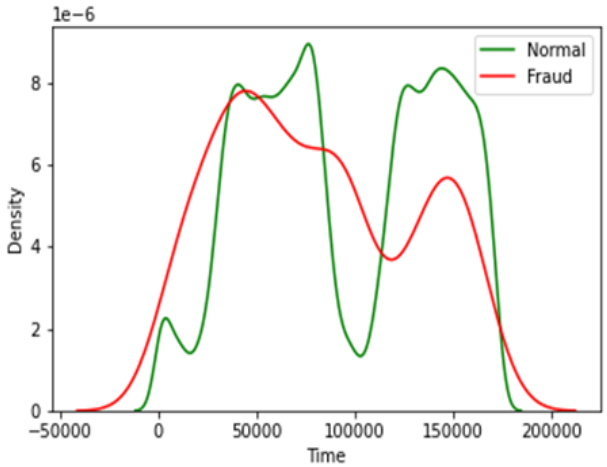

Fig. 1 Distribution of Time for Fraudulent and Non-Fraudulent (Normal) Transactions

C. Proposed Framework

The main idea of this paper is to use recent past transaction data to train a machine learning model and calculate a prediction interval for the transaction amount based on the features in the dataset. The argument is that, if an incoming transaction amount does not fall inside the prediction interval calculated for that transaction by the best fitted model, that particular transaction amount is not described accurately by its features. Therefore, this gives a reason to flag that transaction as potentially fraudulent. Hence, the proposed methods mainly revolve around the concept of calculating prediction intervals for machine learning models. The machine learning models experimented in this paper are ordinary least squares regression, random forest model and the dropout neural network. Fig. 2 provides an overview of the proposed methodology.

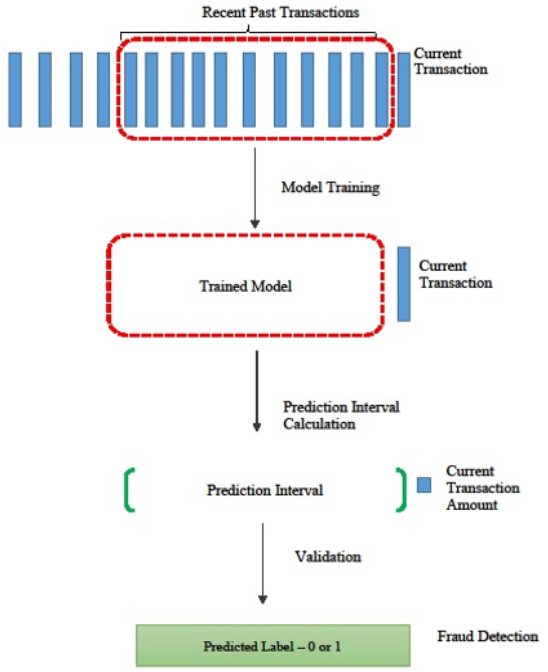

Fig. 2 Overview of the Proposed Fraud Detection Framework

The next section describes the methodologies underlying the different prediction intervals calculated for each of the machine learning models employed in the research.

1. Ordinary Least Squares

Regression is the study of the conditional distribution of the response variable, given a set of features. The normal error regression model is expressed as, $\mathbf{y} = f(X; \boldsymbol{\beta}) + \boldsymbol{\xi}$, where $\mathbf{y}$ is an $n \times 1$ vector of response variables, X is an $n \times p$ matrix of features, and $\boldsymbol{\xi}$ is an $n \times 1$ vector of unknown errors of the model. The term, $f(X; \boldsymbol{\beta})$ is some unknown function defined by $\boldsymbol{\beta}$, the set of parameters of the regression model. It is assumed that the errors, $\boldsymbol{\xi}$, are independent and identically distributed as Gaussian with zero mean and constant variance, σ^2. The ordinary least squares estimate of the parameters $\boldsymbol{\beta}$ is $\widehat{\boldsymbol{\beta}}$ and is obtained by minimizing the objective function, $\max_{\boldsymbol{\beta}} \| \mathbf{y} - f(X; \widehat{\boldsymbol{\beta}}) \|_F^2$, where $f(X; \widehat{\boldsymbol{\beta}}) = \hat{y}$ is the prediction from the model. Under regularity conditions, much of the inference for OLS regression is valid when the assumption on the error distribution is not violated. Under this assumption, $(1-\alpha)100\%$ prediction interval for the unknown response y_* of a new observation with features x_* is given by,

$$\hat{y}_* \pm t_{\left(\frac{\alpha}{2}, n-p\right)} \hat{\sigma} \hat{h}, \qquad (1)$$

where $\hat{y}_*$ is the point prediction, $t_{\left(\frac{\alpha}{2}, n-p\right)}$ is the $\frac{\alpha}{2}$ quantile of the t-distribution with $n-p$ degrees of freedom, and $0 < \alpha < 1$. Here, $\hat{\sigma}^2 = \frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is the unbiased estimate for the unknown variance, σ^2, and $\hat{h}$ is the leverage term estimated as $\hat{h} = \sqrt{(1 + x_*^T (X^T X)^{-1} x_*)}$, where X is the $n \times (p+1)$ design matrix, $\begin{bmatrix} 1 & x_1^T \\ . & . \\ . & . \\ 1 & x_n^T \end{bmatrix}$. Although the calculation of the prediction interval (Equation (1)) is quick and straightforward, the validity of the formula depends heavily on the normality assumption on the errors of the model. In the paper [34], a large sample prediction interval calculation method is presented that does not require knowing the error distribution of the regression. They show that for large samples, Equation (1) can be modified as

$$\left(\hat{y}_* + a_n \hat{\xi}_{\alpha/2}, \hat{y}_* + a_n \hat{\xi}_{1-\alpha/2}\right). \qquad (2)$$

Here, $\hat{\xi}_\alpha$ is the α percentile of the residuals with the ith residual calculated as, $\hat{\xi}_i = y_i - \hat{y}_i$ and

$$a_n = \left(1 + \frac{15}{n}\right)\sqrt{\frac{n}{n-p}}\,\hat{h}. \qquad (3)$$

Equation (2) is very similar to the classical prediction interval presented in Equation (1) except that $\hat{\xi}_\alpha$ is used to estimate the error percentiles as a substitute for $t_{\left(\frac{\alpha}{2}, n-p\right)}\hat{\sigma}$. The term $\sqrt{\frac{n}{n-p}}$ is added as a correction factor for using the residuals to estimate the errors, and $\left(1 + \frac{15}{n}\right)$ is used based on results of several simulation experiments. In this paper, this method will be utilized for calculating prediction intervals for the ordinary least squares regression model.

2. Random Forest

The random forest model is a non-parametric statistical learning method [43]. The usual procedure for performing regression with a random forest is as follows:

a. Draw $1,2,\ldots,B$ bootstrap samples of the original data.
b. For $b = 1,2,\ldots,B$
   i. Randomly sample $p_0 (0 < p_0 < p)$ features and grow a regression tree using binary recursive splitting algorithm to select the best split among these features that minimize the splitting criterion, mean squared error [44].
   ii. Calculate predictions. The response $\hat{y}_{*b}$ of a new test observation x_* is the mean of the responses of the samples that are in the same terminal node.

c. The final prediction for x_* is the mean of the B individual tree predictions, $\hat{y}_* = \frac{1}{B}\sum_{b=1}^{B} \hat{y}_{*b}$.

Since each regression tree is built on its own bootstrap sample of the original data, some observations will not be used for growing a particular tree. Such observations are considered *out of bag* for that particular tree. In this paper, the ideas from [37] and [35] is employed for building a prediction interval for random forest model. The proposed methodology is as follows:

1. Build a forest on the training sample.
2. For each observation with features x_i in the training sample of n observations, identify the regression trees that are out of bag and calculate the mean prediction of that observation's response, $\hat{y}_{(-i)}$ using only those regression trees.
3. Calculate the empirical distribution of out of bag prediction errors, $D_1, \dots, D_n$, where $D_i = \hat{y}_{(-i)} - y_i$ for $i = 1, \dots, n$.
4. It is shown that for large n and large B, the $(1-\alpha)100\%$ prediction interval for the unknown response y_* of a new observation with features x_* can be reasonably assumed as

$$\left[\hat{y}_* - D_{\left(\frac{\alpha}{2},n\right)}, \hat{y}_* + D_{\left(1-\frac{\alpha}{2},n\right)}\right], \tag{4}$$

where $D_{\left(\frac{\alpha}{2},n\right)}$ is the $\frac{\alpha}{2}$ quantile of the empirical distribution of $D_1, \dots, D_n$.

3. Dropout Neural Network

Dropout technique is used to avoid overfitting in artificial neural networks (ANNs). In this paper, an ensemble of dropout neural networks is utilized to create a prediction distribution and calculate prediction intervals. Let us consider an ANN with $\boldsymbol{L}$ hidden layers with $\boldsymbol{l} \in \{\mathbf{1}, \dots, \boldsymbol{L}\}$ denote the index of each layer. The $\mathbf{1} \times (\boldsymbol{l}-\mathbf{1})$vector $\mathbf{z}^{(l)}$ is the inputs to layer $\boldsymbol{l}$ and the $\mathbf{1} \times \boldsymbol{l}$ vector $\mathbf{y}^{(l)}$ is the output from layer $\boldsymbol{l}$. Note that $\mathbf{y}^{(0)} = \mathbf{x}$ is the set of $\boldsymbol{p}$ features in the dataset and $\mathbf{y}^{(L)} = \hat{\boldsymbol{y}}$ is the predicted response. Parameters $\boldsymbol{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weights and biases at layer $\boldsymbol{l}$. The feedforward operation of an ANN for a single training sample can be described as,

$$\begin{aligned} z_k^{(l+1)} &= \boldsymbol{w}_k^{(l+1)}\boldsymbol{y}^l + b_k^{(l+1)}, \\ y_k^{(l+1)} &= f\left(z_k^{(l+1)}\right), \end{aligned} \tag{5}$$

where k denotes the index for a hidden unit and f is a neural network activation function [45]. In a dropout neural network, the feedforward operation described in Equation (5) is modified to include a vector of independent Bernoulli random variables $\mathbf{r}^{(l)}$ with a probability of a success p as,

$$\begin{aligned} r_k^{(l)} &\sim Bernoulli(p), \\ \tilde{\boldsymbol{y}}^{(l)} &= \boldsymbol{r}^{(l)} \otimes \boldsymbol{y}^{(l)}, \\ z_k^{(l+1)} &= \boldsymbol{w}_k^{(l+1)}\tilde{\boldsymbol{y}}^l + b_k^{(l+1)}, \text{ and} \\ y_k^{(l+1)} &= f\left(z_k^{(l+1)}\right), \end{aligned} \tag{6}$$

where $\otimes$ denotes the element-wise product. Thus, a sub-network is sampled from a larger network and derivatives of the network are backpropagated through the sub-network for learning the parameters of the model.

The interested reader may refer [39] for further details on this backpropagation procedure. The learnt model is used at test time after multiplying each $\mathbf{w}_k^{(l)}$ by the probability $r_k^{(l)}$ as $\mathbf{w}_{\text{test}}^{(l)} = r_k^{(l)}\mathbf{w}^{(l)}$. This model is called the dropout neural network. For each new test observation x_*, the below procedure can be followed for calculating a prediction interval.

1. Carry out $B(\geq 1000)$ number of stochastic forward passes through the dropout neural network and obtain the predctive distribution, $\hat{y}_*^1, \hat{y}_*^2, \dots, \hat{y}_*^B$.
2. Calculate the predictive mean as $\hat{y}_* = \sum_{b=1}^{B} \hat{y}_*^b$.
3. Calculate the standard deviation of the predictive distribution as
$\hat{\sigma}_{\hat{y}_*} = \left[\frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{y}_*^b - \hat{y}_*\right)^2\right]^{1/2}$.
4. By the central limit theorem, the prediction $(1-\alpha)100\%$ prediction interval for the unknown response y_* can be calculated as

$$\hat{y}_* \pm z_{1-\alpha/2}\frac{\hat{\sigma}_{\hat{y}_*}}{\sqrt{B-1}}, \tag{7}$$

where $z_{1-\alpha/2}$ is the $\frac{1-\alpha}{2}$ quantile of the t-distribution.

This technique is formally known as MC Dropout and is presented in [40] for quantifying predictive uncertainty of an ANN. Based on the machine learning models used and the prediction interval calculation methods employed for each method, three unsupervised methods are proposed for fraud detection.

If the machine learning model trained on recent past data is the ordinary least squares regression model, and the prediction interval is calculated using Equation (2), we call the resulting approach as the Ordinary Least Squares-based (OLS) Method. On the other hand, if the random forest model is used to train the data and the prediction interval is calculated using Equation (4), we call it the Random Forest-based (RF) method.

Finally, if the machine learning model employed is the dropout neural network and the prediction interval is calculated using Equation (7), the method is called the Dropout Neural Network-based (DNN) method. Table I summarizes these methods.

TABLE I SUMMARY OF PROPOSED METHODS

Machine Learning Model	Prediction Interval	Fraud Detection Method
Ordinary Least Squares	Equation (2)	OLS
Random Forest	Equation (4)	RF
Dropout Neural Network	Equation (7)	DNN

D. Performance Measure

This paper employs the confusion matrix and three other metrics calculated using the confusion matrix to evaluate the fraud detection performance of our proposed methods. Table II describes a confusion matrix resulting from a binary classification algorithm. Recall from Section III.B, that the labels for the transactions are available for the dataset. Although these labels are not used for training our models, they are used to evaluate the predictions calculated from the models. Thus, the predicted labels are compared against the actual labels available for the test set. The following procedure is used for calculating the performance measures for each model.

1. Feed the unlabeled training dataset to the machine learning model and learn the model.
2. Feed each transaction in the test set to the trained model and calculate a prediction interval for the transaction amount.
3. If the observed transaction amount is inside the prediction interval, flag that transaction as a non-fraudulent transaction, otherwise flag the transaction as a fraudulent transaction. The resulting flags are the predicted labels for each transaction.
4. Compare the predicted labels with the actual labels in the dataset by calculating the performance measures, accuracy (Equation (8)), precision (Equation (9)) and recall (Equation (10)).

This displays the possible outcomes of a binary classification problem. Type I errors are the false positives (FP) and type II errors are the false negatives (FN)

TABLE II CONFUSION MATRIX

Actual	**Predicted**	
	0	1
0	True Negative (TN)	False Positive (FP)
1	False Negative (FN)	True Positive (TP)

Definition 2 (Accuracy). Proportion of correctly predicted samples (fraud or non-fraud) out of all the samples:

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN}. \qquad (1)$$

Definition 3 (Precision). Proportion of correctly detected samples out of the actual fraudulent samples:

$$\text{precision} = \frac{TP}{TP+FP} \qquad (9)$$

Definition 4 (Recall). Proportion of actual fraudulent samples out of the predicted fraudulent samples. Also called the true positive rate:

$$\text{recall} = \frac{TP}{TP+FN} \qquad (2)$$

E. Comparison Methods

In this paper, the performances of proposed three methods are compared against two different state-of-the-art algorithms that also detect fraudulent transactions in an unsupervised setting.

1. Local Outlier Factor: The Local Outlier Factor (LOF) algorithm is introduced in [22] to detect anomalous observations in a dataset. *Locality* captures the k nearest neighbours surrounding a particular observation, and local density captures the average distances between them. Based on the local density, the LOF algorithm calculates a score for each observation that measures the amount of isolation of a point concerning its neighbours. By comparing the calculated LOF measure to the lower and upper bound, an observation is classified as an anomaly.

2. Isolation Forest

The Isolation Forest (IF) algorithm, proposed in [21] works on the objective of isolating anomalous observations in the dataset. The underlying basis is a decision tree algorithm that recursively partitions the observations until all are isolated. This random partitioning produces comparatively shorter path lengths in the tree structure for anomalies. An anomaly score is calculated for each observation by considering its path lengths in the overall forest of random trees. Those observations that possess shorter path lengths receive higher anomaly scores and are classified as anomalies based on a thresholding method that is also introduced in the same paper [21].

IV. EXPERIMENTAL RESULTS

As discussed in Section III.B, the transaction dataset considered in this paper consists of 28 features, and the transaction amount is considered as the response variable. A percentage of 90% of the samples is allocated for the train set and the rest of the 10% for the test set. Based on our terminology, the train set represents recent past data, and the test set is the current data. As previously described in Fig.2, recent past data is used for training the machine learning model. Current data represents new data that needs to be validated. This validation is done by applying the trained model to the current data, estimating prediction intervals and flagging those samples that fall outside the interval as fraudulent. As discussed in Section III.D, if a transaction amount falls outside its prediction interval, the label will be predicted as fraudulent and vice versa.

A. OLS Method

This paper uses Python's Scikit learn package to implement the ordinary least squares regression model. The main assumption of ordinary least squares regression is that the features are not correlated. Since the 28 features of the experimented dataset are principal components calculated using the original features (Section III.B), this assumption is not violated. Our ordinary least squares model fitted on the training set shows an R-squared value of 0.92. The value of 0.92 suggests a good fit for the data. Then as described in Section III.C.1, the model parameters learned from the training set are then applied to the test set to calculate a prediction interval for each test sample and predict the labels.

B. RF Method

The Random Forest Regressor algorithm in Python's Scikit learn package provides a range of hyper-parameter options to fit a random forest on a dataset to improve the accuracy of the prediction. Scikit-Learn's Grid Search CV method is used for hyper-parameter tuning with cross-validation to select the best combination of hyper-parameters that provide an optimal result. A forest with 1000 regression trees is built on the training data, and out-of-bag errors are calculated for each training sample. The trained model is then examined on the test data to predict the transaction amount for each test sample. Using the distribution of out-of-bag errors from the training sample, prediction intervals are calculated for each test sample as explained in Section III.C.2 and predict the labels.

C. DNN Method

The Keras package in Python is used to construct the dropout neural network. Like the previous two methods, the model is first trained on the training set. A simple four-layer dense neural network is used to model the data. The inputs are transferred into 28 nodes, then transferred into two hidden layers, each with 64 units. The final layer with a single node is used to predict the output. Two dropout layers are implemented after the dense hidden layers in the middle. Here, 50 per cent of the hidden units are dropped per layer. In the compilation section of the model, the mean square error (MSE) is set as the loss function, while adam is set as the optimizer. The tangent (tanh) activation function converts the inputs to outputs at each hidden unit. Although there are other activation functions, such as Relu and sigmoid, they are unsuitable for this dataset where the features can contain negative values. The Relu function converts negative values to zero, disrupting the learning of gradient information, while the sigmoid function causes the vanishing of the gradient. After training the model, the learned dropout neural network is fitted over 1000 runs on the test set to obtain the predictive distribution for each test sample.

Then, using the procedure described in Section III.C.3 the prediction interval is calculated for each test sample. After calculating the prediction intervals for the test samples, the standard approach described in step 3 of Section III.D is used by all three methods to validate transactions and flag the samples as fraudulent or non-fraudulent.

D. IF and LOF Methods

The IF and LOF methods are implemented using the Isolation Forest algorithm and the Local Outlier Factor algorithm in Python, respectively. In addition to the 28 features used as input to OLS, RF and DNN methods, the transaction amount is also included as an input feature to IF and LOF algorithms. Each of these algorithms discover the anomalies in the dataset. These anomalies are considered as the predicted fraudulent transactions from these methods.

E. Performance Comparison

The predicted labels for each method are compared with the actual labels available for the dataset to calculate the performance measures. Table III presents the confusion matrices obtained from the three methods and the two state-of-the-art methods.

TABLE III CONFUSION MATRICES OF DIFFERENT MODELS

OLS			**RN**			**DNN**			**IF**			**LOF**		
Actual	Predicted		Actual	Predicted		Actual	Predicted		Actual	Predicted		Actual	Predicted	
	0	1		0	1		0	1		0	1		0	1
0	26985	1447	0	26895	1537	0	2697	25735	0	283925	390	0	283865	450
1	32	17	1	34	15	1	3	46	1	329	163	1	343	149

Table III and Fig. 3 presents the performance metrics, accuracy, precision and recall for each method. In terms of accuracy, the OLS method, with an accuracy rate of 95% and the RF method, with an accuracy rate of 94%, have performed well in correctly predicting the label of a transaction. However, still, OLS and RF methods do not outperform the IF or LOF method in terms of the accuracy of the predictions. Although the DNN method has a low accuracy level of 9%, it is observed that it predicts a majority of samples as fraudulent. This shows that the method is very sensitive to smaller changes in the behaviour of the transactions.

With a high recall value of 94%, the DNN method is the most reliable method for capturing actual fraudulent transactions. The recall rates of OLS and RF methods are 35% and 31%, respectively. Regarding Recall, the OLS and RF methods perform similarly to the state-of-the-art methods. Thus, the DNN method shows the highest capability to capture fraudulent transactions. In terms of precision, all methods do not show satisfactory results. This shows that all unsupervised methods show a considerably high false-positive rate. However, out of the three proposed methods, the OLS method shows comparatively higher precision than the other two methods. For fraud detection, a method having a high recall rate, but a low precision rate can still be very useful because missing a fraudulent transaction is much costlier than a false detection. Thus, although the DNN method has a low precision rate, it can still be considered as an effective method. From the overall results, the OLS method and the RF method show the best performance in correctly identifying whether a transaction is fraudulent or

non-fraudulent. The DNN method shows the highest reliability in detecting fraudulent transactions. The 94% recall rate shows that there is only a 6% chance that a fraudulent transaction will be missed by the DNN method. Although the proportion of fraudulent transactions in the dataset is very small, the DNN method is not affected by this class imbalance problem.

TABLE IV DETECTION PERFORMANCE OF METHODS

Method	Accuracy	Precision	Recall
OLS	0.9481	0.0116	0.3469
RF	0.9448	0.0097	0.3061
DNN	0.0963	0.0018	0.9388
IF	0.9974	0.2948	0.3313
LOF	0.9972	0.2487	0.3028

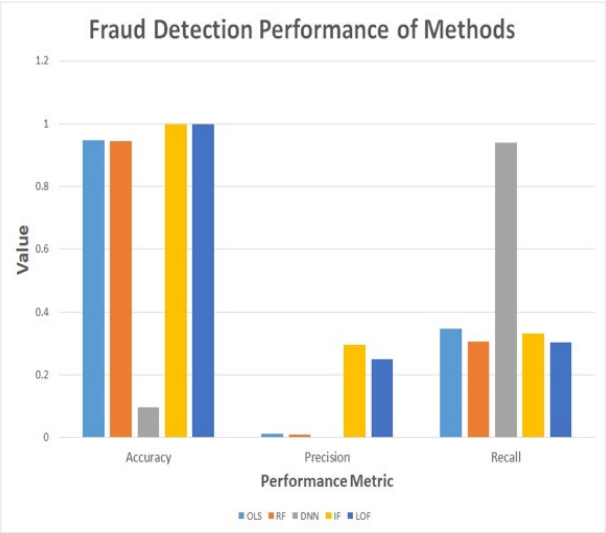

Fig. 3 Performance Comparison of Fraud Detection Performance

V. CONCLUSION AND FUTURE WORK

The main goal of this paper is to propose three unsupervised online fraud detection methods, OLS, RF and DNN, to validate incoming transactions and predict if they are fraudulent or non-fraudulent. Considering the transaction amount as the response, and other transaction attributes as features, for each method, a machine learning model is trained on the recent past transaction data. Then, the learned model is applied to an incoming transaction to calculate a prediction interval for the transaction amount. If the actual transaction amount lies outside the predicted interval, that transaction is flagged as fraudulent. Since the idea of applying prediction intervals for fraud detection is relatively new and not much research exists, we apply machine learning models of various spectrums and check their efficiency for the said task. The models employed include the ordinary least squares regression model, random forest model and the dropout neural network model. Based on the machine learning model and the prediction interval calculation equation, each approach defines a unique unsupervised method for fraud detection. Our methods are unsupervised, as the transaction labels are not required to train the machine learning models. Although the dataset employed in this research is rich with label information, they are not utilized for training the models. Instead, these labels are employed to evaluate the predictions by calculating the accuracy, precision and recall measures for the predictions. From the overall results, the OLS and RF methods, with accuracy rates of more than 94%, generally perform well in predicting whether a transaction is fraudulent or non-fraudulent. The DNN method, with a high recall rate of 94%, is the most robust method for detecting those fraudulent transactions in the dataset.

We propose a novel unsupervised approach of prediction intervals for fraud detection in transaction data. In the evaluated dataset, only 0.172% of the transactions are fraudulent. Thus, there is an enormous class imbalance. However, still, the performance of the proposed methods is evaluated without addressing the class imbalance. This is purposely done because our goal is to introduce a novel method that can be applied to real-world applications. In the real world, fraud labels are not available, so it is not practically possible to address the class imbalance problem. As the proposed methods are in an early phase, the performance may not be highly significant, but there are many exciting avenues for further investigation. The current research employs features previously transformed into principal components due to confidentiality reasons. However, if customer information and transaction details are available, some feature engineering strategies as discussed in [13] can be employed to incorporate each customer's usual transaction patterns further to enhance the performance of the proposed fraud detection methods. In this study, we discuss three different methods of calculating prediction intervals for the three models experimented. An interesting extension would be to investigate a unified approach for calculating a prediction interval for the transaction amount predicted by any machine learning model trained for the data. For this purpose, nonparametric methods such as kernel density functions [46] and quantile regression methods [47] may be suitable.

Furthermore, the overall effectiveness of the proposed methods will be primarily influenced by the confidence level defined for the interval calculation. Conducting a systematic simulation to decide the optimum confidence level would be another future research development. It is observed that accuracy is high for the OLS and RF methods, while recall is high for the DNN method. A method that combines the predictions of all three methods, such as a soft voting classifier as discussed in [48] would be worth investigating. Due to the huge loss incurred by many organizations, detecting fraudulent transactions is considered as one of the most investigated areas of economic crime. Although there exist many supervised classification methods for fraud detection, it becomes problematic when applying these methods in real-world problems as no label information on transactions is available. Furthermore, as fraudsters frequently adapt their behaviour to stay undetected, it is

important to continuously update the fraud detection algorithm using data on recent past transactions. In this paper, the issues are addressed, and three unsupervised methods are proposed for fraud detection that can be trained online. The dataset used in this study is available at https://www.kaggle.com/mlg-ulb/creditcard fraud.

REFERENCES

[1] M. Zanin, M. Romance, S. Moral, and R. Criado, "Credit card fraud detection through parenclitic network analysis," *Complexity*, Vol. 2018, 2018.

[2] E. W. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decision support systems*, Vol. 50, No. 3, pp. 559-569, 2011.

[3] J. West and M. Bhattacharya, "Intelligent financial fraud detection: A comprehensive review," *Computers & security*, Vol. 57, pp. 47-66, 2016.

[4] M. Zareapoor, K. Seeja, and M. A. Alam, "Analysis on credit card fraud detection techniques: Based on certain design criteria," *International journal of computer applications*, Vol. 52, No. 3, 2012.

[5] V. Zaslavsky and A. Strizhak, "Credit card fraud detection using self-organizing maps," *Information and Security*, Vol. 18, pp. 48, 2006.

[6] C. Liu, Y. Chan, S. H. Alam Kazmi, and H. Fu, "Financial fraud detection model: Based on random forest," *International journal of economics and finance*, Vol. 7, No. 7, 2015.

[7] E. M. Carneiro, L. A. V. Dias, A. M. Da Cunha, and L. F. S. Mialaret, "Cluster analysis and artificial neural networks: A case study in credit card fraud detection," in *2015 12th international conference on information technology-new generations*, pp. 122-126, 2015.

[8] D. Excell, "Bayesian inference–the future of online fraud protection," *Computer Fraud & Security*, Vol. 2012, No. 2, pp. 8-11, 2012.

[9] D. J. Hand, "Discrimination and classification," *Wiley Series in Probability and Mathematical Statistics*, 1981.

[10] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE transactions on neural networks and learning systems*, Vol. 29, No. 8, pp. 3784-3797, 2017.

[11] Y. Sahin and E. Duman, "Detecting credit card fraud by ANN and logistic regression," in *2011 international symposium on innovations in intelligent systems and applications*, pp. 315-319, 2011.

[12] S. V. Suryanarayana, G. Balaji, and G. V. Rao, "Machine learning approaches for credit card fraud detection," *Int. J. Eng. Technol*, Vol. 7, No. 2, pp. 917–920, 2018.

[13] A. C. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, "Feature engineering strategies for credit card fraud detection," *Expert Systems with Applications*, Vol. 51, pp. 134-142, 2016.

[14] A. Bach, "Boltzmann's probability distribution of 1877," *Archive for History of Exact Sciences*, Vol. 41, No. 1, pp. 1-40, 1990.

[15] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[16] C. Elkan, "The foundations of cost-sensitive learning," in *International joint conference on artificial intelligence*, Vol. 17, pp. 973–978, 2001.

[17] A. Srivastava, A. Kundu, S. Sural, and A. Majumdar, "Credit card fraud detection using hidden markov model," *IEEE Transactions on dependable and secure computing*, Vol. 5, No. 1, pp. 37-48, 2008.

[18] M. Rezapour, "Anomaly detection using unsupervised methods: Credit card fraud case study," *Int J Adv Comput Sci Appl*, Vol. 10, No. 11, 2019.

[19] R. J. Bolton, D. J. Hand, *et al.*, "Unsupervised profiling methods for fraud detection," *Credit scoring and credit control VII*, pp. 235-255, 2001.

[20] M. R. Lepoivre, C. O. Avanzini, G. Bignon, L. Legendre, and A. K. Piwele, "Credit card fraud detection with unsupervised algorithms," *Journal of Advances in Information Technology*, Vol. 7, No. 1, 2016.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth IEEE international conference on data mining*, pp. 413-422, 2008.

[22] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on management of data*, pp. 93-104, 2000.

[23] H. John and S. Naaz, "Credit card fraud detection using local outlier factor and isolation forest," *Int. J. Comput. Sci. Eng*, Vol. 7, No. 4, pp. 1060-1064, 2019.

[24] Z. Cheng, C. Zou, and J. Dong, "Outlier detection using isolation forest and local outlier factor," in *Proceedings of the conference on research in adaptive and convergent systems*, pp. 161-168, 2019.

[25] A. K. Shaikh and A. Nazir, "A novel dynamic approach to identifying suspicious customers in money transactions," *International Journal of Business Intelligence and Data Mining*, Vol. 17, No. 2, pp. 143-158, 2018.

[26] K. Shaikh and A. Nazir, "A model for identifying relationships of suspicious customers in money laundering using social network functions," in *Proceedings of the world congress on engineering*, Vol. 1, pp. 4-7, 2018.

[27] R. Krzysztofowicz and K. S. Kelly, "Hydrologic uncertainty processor for probabilistic river stage forecasting," *Water resources research*, Vol. 36, No. 11, pp. 3265-3277, 2000.

[28] K. Beven and A. Binley, "The future of distributed models: Model calibration and uncertainty prediction," *Hydrological processes*, Vol. 6, No. 3, pp. 279-298, 1992.

[29] S. Maskey, V. Guinot, and R. K. Price, "Treatment of precipitation uncertainty in rainfall-runoff modelling: A fuzzy set approach," *Advances in water resources*, Vol. 27, No. 9, pp. 889-898, 2004.

[30] S. Kumar and A. Srivastava, "Bootstrap prediction intervals in non-parametric regression with applications to anomaly detection," 2012.

[31] B. Lu and J. Hardin, "A unified framework for random forest prediction error estimation," *Journal of Machine Learning Research*, Vol. 22, No. 8, pp. 1-41, 2021.

[32] J. J. Faraway, *Practical regression and ANOVA using R.*, Vol. 168, Citeseer, 2002.

[33] R. A. Stine, "Bootstrap prediction intervals for regression," *Journal of the American Statistical Association*, Vol. 80, No. 392, pp. 1026-1031, 1985.

[34] D. J. Olive, "Prediction intervals for regression models," *Computational statistics & data analysis*, Vol. 51, No. 6, pp. 3115-3122, 2007.

[35] H. Zhang, J. Zimmerman, D. Nettleton, and D. J. Nordman, "Random forest prediction intervals," *The American Statistician*, 2019.

[36] N. Meinshausen and G. Ridgeway, "Quantile regression forests." *Journal of Machine Learning Research*, Vol. 7, No. 6, 2006.

[37] B. Lu and J. Hardin, "Constructing prediction intervals for random forests," PhD thesis, Pomona College, 2017.

[38] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *Journal of the American Statistical Association*, Vol. 113, No. 523, pp. 1094-1111, 2018.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, Vol. 15, No. 1, pp. 1929-1958, 2014.

[40] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International conference on machine learning*, pp. 1050-1059, 2016.

[41] A. Damianou and N. D. Lawrence, "Deep gaussian processes," in *Artificial intelligence and statistics*, pp. 207-215, 2013.

[42] M. Rosenblatt, "A central limit theorem and a strong mixing condition," *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 42, No. 1, pp. 43, 1956.

[43] L. Breiman, "Random forests," *Machine learning*, Vol. 45, No. 1, pp. 5-32, 2001.

[44] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random forests," in *Ensemble machine learning*, Springer, 2012, pp. 157-175.

[45] S. Sharma and S. Sharma, "Activation functions in neural networks," *Towards Data Science*, Vol. 6, No. 12, pp. 310-316, 2017.

[46] L. J. Latecki, A. Lazarevic, and D. Pokrajac, "Outlier detection with kernel density functions," in *International workshop on machine learning and data mining in pattern recognition*, pp. 61-75, 2007.

[47] K. Yu and M. Jones, "Local linear quantile regression," *Journal of the American statistical Association*, Vol. 93, No. 441, pp. 228-237, 1998.

[48] M. Rout *et al.*, "Analysis and comparison of credit card fraud detection using machine learning," in *Advances in electronics, communication and computing, Springer,* pp. 33-40, 2021.