Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 2439205, 12 pages
https://doi.org/10.1155/2022/2439205

Research Article

An Innovative Sensing Machine Learning Technique to Detect Credit Card Frauds in Wireless Communications

G. Sasikala,[1] M. Laavanya,[2] B. Sathyasri,[1] C. Supraja,[1] V. Mahalakshmi,[1] S. S. Sreeja Mole,[3] Jaison Mulerikkal,[4] S. Chidambaranathan,[5] C. Arvind,[6] K. Srihari,[7] and Minilu Dejene[8]

[1]*Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, Tamil Nadu, India*
[2]*Department of Electronics and Communication Engineering, Vignan's Foundation for Science, Technology and Research, Vadlamudi, Guntur, Andhra Pradesh 522213, India*
[3]*Department of ECE, Christu Jyothi Institute of Institute of Technology and Science, Yeswanthapur, Jangaon 506167, India*
[4]*Department of Information Technology, Rajagiri School of Engineering and Technology, Kochi, 682039 Kerala, India*
[5]*Department of Computer Applications, St. Xavier's College (Autonomous), Palayamkottai, 627002 Tamil Nadu, India*
[6]*Department of ECE, Karpagam College of Engineering, Coimbatore, Tamil Nadu, India*
[7]*Department of CSE, SNS College of Technology, Coimbatore, Tamil Nadu, India*
[8]*Department of Biotechnology, College of Biological and Chemical Engineering, Addis Ababa Science and Technology University, Addis Ababa, Ethiopia*

Correspondence should be addressed to K. Srihari; hariontо@gmail.com and Minilu Dejene; minilu.dejene@aastu.edu.et

Received 15 February 2022; Revised 13 April 2022; Accepted 20 April 2022; Published 23 June 2022

Academic Editor: Mohammad Farukh Hashmi

There has been an increase in credit card fraud as e-commerce has become more widespread. Financial transactions are essential to our economy, so detecting bank fraud is essential. Experiments on automated and real-time fraud detection are needed here. There are numerous machine learning techniques for identifying credit card fraud, and the most prevalent are support vector machine (SVM), logic regression, and random forest. When models penalise all errors equally during training, the quality of these detection approaches becomes crucial. This paper uses an innovative sensing method to judge the classification algorithm by considering the misclassification cost and at the same time by employing SVM hyperparameter optimization using grid search cross-validation and separating the hyperplane using the theory of reproducing kernels like linear, Gaussian, and polynomial, and the robustness is maintained. Because of this, credit card fraud has been identified significantly more successful than in the past.

1. Introduction

Credit card fraud is increasing in popularity as a result of the prevalence of online purchasing. According to Robertson's calculations, credit card fraud losses climbed dramatically between 2010 and 2015, from $7.6 billion to $21.81 billion [1]. Credit card fraud loss globally in 2020 is $31.67 billion. Credit card fraud may be perpetrated by criminals who get access to your personal data. When it comes to your credit card, there are a variety of methods that fraudsters may get their hands on your personal information, credit card number, CVV, and one-time password (OTP). Fraudsters and scammers frequently use the following methods to commit credit card fraud:

(i) Physical access to the credit card

Theft of a credit card is the most typical method of gaining access to your personal information.

(ii) Skimming your credit card

The little skimming devices linked to the point of scale equipment might steal our credit card information. The skimming machine has already scanned and stored your credit card information when you swipe it through a POS terminal. While the skimming device obtains your credit card information, a concealed camera nearby records your PIN, which criminals may use to complete the transaction.

(iii) Phishing

Phishing is nothing more than pretending to be a reputable firm and requesting that the victim click on a link that appears to be a valid one.

(iv) Malware Attack

A malware attack occurs when a hacker or fraudster infects a victim's computer without the victim's knowledge and installs malicious software without the victim's knowledge.

(v) Stealing your email details

(vi) Stealing your documents

Today's fraud detection methods are hopelessly ineffective, resulting in huge financial losses for both merchants and card companies. As fraud detection technology improves, fraudsters are always increasing their capacity to elude detection. Machine learning is a subfield of artificial intelligence (AI) and computer science that focuses on using data and algorithms to mimic how people learn, progressively improving its accuracy. Supervised learning, unsupervised learning, and reinforcement learning are the three types of learning algorithms.

Supervised learning is a machine learning method distinguished by the use of labelled datasets. The model may test its accuracy and learn over time by using labelled inputs and outputs. Supervised learning may be applied to classification and regression problems. Unsupervised learning problems make use of unlabelled training data to model the data's underlying structure. It is employed in the fields of association, clustering, and dimensionality reduction. Reinforcement learning is a form of machine learning technique that enables the agent to determine the optimum future action based on its current state by learning behaviours that maximise the reward. Typically, reinforcement learning systems learn optimum actions through trial and error. They are commonly seen in robotics and video games.

Unsupervised and supervised methods of detecting credit card fraud are the two broad classifications. Supervised fraud detection is a technique for identifying fraudulent transactions by analysing a sample of both normal and suspicious transactions [2]. Unsupervised fraud detection looks for unusual or out-of-the-ordinary transactions that might be fraudulent. The likelihood of a transaction being fraudulent can be predicted using detection techniques such as SVM [3], logistic regression [4], random forest [5], and Naïve Bayes [6]. These algorithms are frequently employed in the detection of credit card fraud. Logistic regression may be used to estimate the likelihood of a target variable occurring. In a dichotomous variable, there are only two potential classes of the target or dependent variable. Dependent variables may only be categorised as either 1 or 0 (success/yes or failure/no); hence, the dependent variable is binary in nature [4, 7]. On the other hand, random forest uses a collection of decision trees that have been trained using the "bagging" approach. The bagging approach is based on the concept that a mixture of learning models would yield a better overall outcome [8, 9].

When models penalise all misclassifications with the same penalty during training, the reliability of these detection techniques becomes more important. Hence, the cost of misclassification must be taken into account while developing new approaches for detecting credit card fraud.

SVM is a form of machine learning approach that can be used to solve problems related to data classification. Image recognition [10], credit scoring [11], public safety [12], and classification [13–15] are just a few of the many fields in which it is used. In recent years, the usage of DNN for credit card fraud detection has increased [16]. A few of them are autoencoder-based detection [17], K-means deep network [18], and LSTM using attention mechanism dependence concerns are better addressed by RNN architecture. Ability to learn order dependency in sequence prediction problems as a behaviour required in complicated issue areas like as machine translation and speech recognition, among others [19]. But when dealing greater than 2D, noisy input data, the support vector machine's classification performance is significantly lower than when dealing with less than 2D, clean data. The noise can be removed by autoencoder [20–22].

This study's major goal is to evaluate in depth how effective support vector machines are at sensing credit card fraud. The SVM is superior to other classifiers in that it is capable of separating data either by a hyperplane and by use of kernels. This can be done by either looking for a straight line that connects the data or moving the data into a high-dimensional space that makes it clear which data comes from which support vector. The settings of the kernel function influence the classification performance of the SVM [23]. To find parameters, the SVM primarily makes use of the random and grid search approaches [24]. In random approach, random combinations are selected but in the grid search approach, we select the combinations. The smaller the dataset for random search, the faster but less precise the optimization. The richer the dataset, the more precise the optimization, but the closer the optimization is to a grid search. A key disadvantage of the random search method is that it does not use earlier trials' data to choose the next batch, nor does it employ a technique to forecast the next trial. Grid search is a time-honoured technique that is used for all combinations. Cross-validation extracts the majority of the dataset's patterns and trains the model for each combination. During the validation phase, it produces the combination with the highest performance. The disadvantage is that it frequently takes longer for high dimension

[25–27]. In grid search, parameters are not perfect because it is quite easy to get stuck in the optimal solution within the neighbouring set of data. In this article, the hyperparameters of the SVM are changed using a grid search cross-validation search technique while taking into consideration the cost of misclassification in the SVM. This sensing machine learning technique helps to avoid stuck in local optimums. In other words, the proposed approach performs better in categorisation.

There are three main components to the paper. SVMs that account for the misclassification costs associated with credit card fraud detection will be developed in Section 2. Experiments will be carried out in Section 3, and the study's findings and recommendations will be presented in Section 4.

2. Data Classification using SVM

Support vector machines are supervised learning models that can be used for both classification and regression. SVM is a frequently used machine learning algorithm due to its versatility and ease of usage. With SVM, we can easily assign new data points to the correct category of hyperplane by categorising n-dimensional space into classes, as shown in Figure 1. SVM is part of the field of supervised machine learning. The support vector classifier (SVC) algorithm is used if the hyperplane classifies data linearly. A nonlinear strategy to separate the dataset is used by SVM. Graphical representations of linear and nonlinear data separation are shown in Figures 2(a) and 2(b). Two unique datasets are separated by a line and a squiggle in both of the graphs.

In general, SVM can be used for condition monitoring and defect diagnosis [28, 29]. Known as the "kernel trick," SVM makes use of this method [23]. Low-dimensional input spaces can be transformed into higher dimensions by these functions, i.e., they transform unsolvable problems into solvable problems, or vice versa. According to [30], in kernel function Gaussian radial basis function (RBF), the parameter sigma needs correct adjustment for reliable SVM performance. Setting up an SVM is like solving the quadratic optimization problem to find the thinnest hyperplane across classes. The number of support vectors has a direct correlation to the number of characteristics that have been addressed in [31, 32].

The training data consists of a set of points (vectors) x_j and the categories that they fall into y_j. For a given dimension d, the $x_i \epsilon R^d$, and the $y_j = \pm 1$. The hyperplane is

$$f(x) = x'\beta + b = 0. \tag{1}$$

In equation (1), $\beta \epsilon R^d$ and the best separating hyperplane is b. $\|\beta\|$ can be minimized by taking into account all data points (x_i, y_j) and by finding β and b.

$$y_j f(x_j) \geq 1. \tag{2}$$

All of the x_i on the boundary are support vectors. For each of them, the support vectors are $y_j f(x_j) = 1$. This is a quadratic programming problem. z can be classed as a vector defined in equation (3) as a result of the optimal solution $(\widehat{\beta}, \widehat{b})$.

$$\text{Class}(z) = \text{sign}\left(z'\,\widehat{\beta} + \widehat{b}\right) = \text{sign}\left(\widehat{f}(z)\right). \tag{3}$$

In equation (3), $\widehat{f}(z)$ is a classification score, and it indicates the distance between z and decision boundary. When the data cannot be separated into distinct groups, SVM employs a soft margin. Soft margins can be created in two ways: through a penalty parameter C and the use of slack variable ξ_j. This implies the existence of a hyperplane that divides a substantial number of data points but not all of them.

The L^1-norm problem is $\min_{\beta b \xi}(1/2\widehat{\beta}, \beta + C\sum_j \xi_j)$, such that

$$\begin{gathered} yjf(xj)\xi j \geq 1 - \xi j, \\ \xi_j > 0. \end{gathered} \tag{4}$$

Instead of calculating the squares of L^1-norm, it suggests using them as slack variables. The L^1-norm problem is solved using Sequential Minimal Optimization (SMO). The constraints that are applied to L^1-norm are also applied to L^2-norm.

The slack variables ξ_j are given more weight in these formulations, as C increases, i.e., setting a big C results in a high penalty for misclassification, whereas setting a low C results in a low penalty for misclassification. Basic hyperplanes are not always successful in binary classification situations. For these challenges, a mathematical solution that leverages the notion of reproducing kernels retains practically all of the simplicity of an SVM splitting hyperplane is at hand.

It operates by taking into account S and φ to map to S.

$$G(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle. \tag{5}$$

In most cases, samples that cannot be separated linearly are the exception rather than the rule. The linear kernel is shown to be represented as in the following equation:

$$K(x_1, x_2) = {x_1}^T x_2. \tag{6}$$

It is possible to enhance the prediction of SVM when the classification problem is not linearly separable. By mapping the data in to high-dimensional space, nonlinear feature separation is possible [4]. The Gaussian RBF kernel is an example of a nonlinear kernel. It is defined by

$$G(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2)\|^2}{2\sigma^2}\right), \tag{7}$$

where σ denotes the kernel's width. If the parameter is near 0, the SVM is unable to fit new data, indicating that it is too

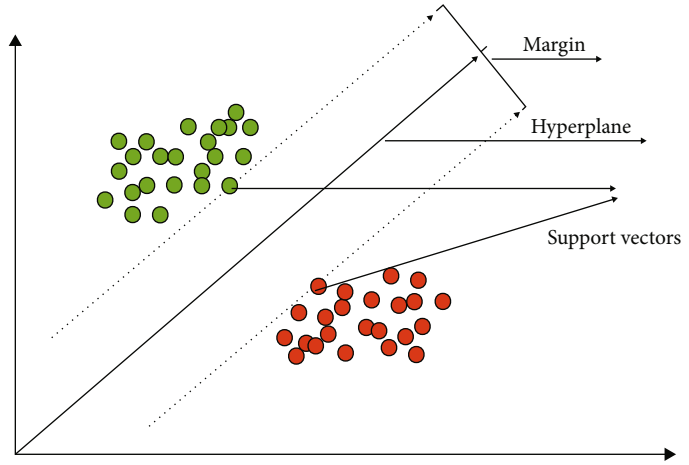

Figure 1: SVM dimensional space.

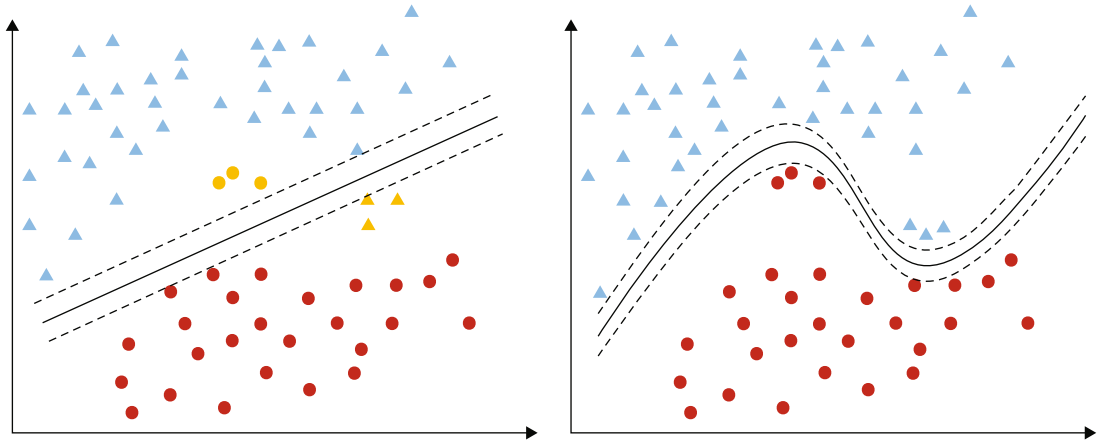

Figure 2: (a) Linear data separation. (b) Nonlinear data separation.

close to support vectors. By increasing the value of, all occurrences are classified as underfitting.

Polynomial is the frequently used kernel function defined as

$$K(x_1, x_2) = \left(x_1{}^T x_2 + 1\right)^p. \tag{8}$$

The order of the polynomial kernel is p. The linear kernel is the polynomial with the lowest degree, and it is not used when the features have a nonlinear relationship. The degree of the polynomial kernel determines how flexible the classifier is, and higher-degree polynomials allow for more flexible decision limits than linear ones [23].

2.1. Decision Boundary and Ordering of SVM. A key objective of the SVM method is to find the optimum decision boundary or line that can divide n-dimensional space into classes, allowing us to quickly classify fresh data points in the future. We need to identify the optimal decision boundary to categorise the data points in n-dimensional space, even if there are several lines/decision boundaries. It is called the hyperplane of SVM since it is the most optimal boundary.

To create the hyperplane, the SVM uses the most extreme points/vectors in the dataset. Support vectors refer to extreme instances. As a result of a change in misclassification cost, we alter the ordering since the orientation of the hyperplane often shifts. ROC curves do not change when the separating hyperplane is translated in feature space, but when it is rotated. As a result, great care must be given when determining the cost of incorrect categorisation.

2.2. Assigning Data Points to the Hyperplane. Support vectors are the data points closest to the decision border; they are the most difficult to categorise data points, and they are critical for SVM to be the ideal decision surface. The purpose of this hyperplane is to be as far away from the support vectors as possible. Margin is the distance between hyperplanes and support vectors. Thus, the optimal hyperplane is the one with the largest margin.

However, in real-world applications, the amount of data that overlaps is so great that soft margin SVMs are incapable of producing an appropriate classifier. As an alternative,

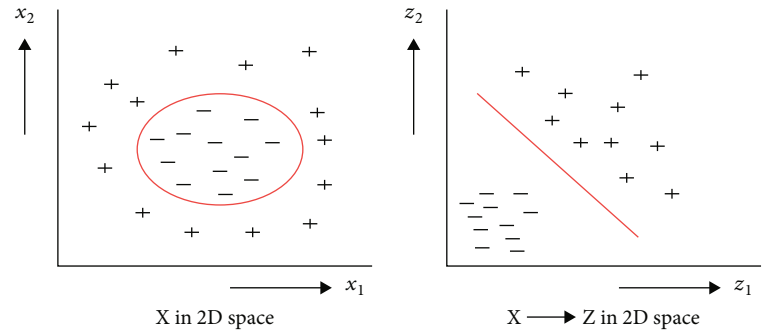

Figure 3: Hyperplane mapping from nonlinear to linear.

there is a requirement to compute a nonlinear decision boundary (i.e., not a hyperplane rather hypersurface). It is worth noting that a linear hyperplane is defined as a linear equation in terms of the n-dimensional component, whereas a nonlinear hypersurface is defined as a nonlinear expression. A hyperplane is defined as follows:$c = w1x1 + w2x2 + w3x3$ while a nonlinear hypersurface is defined as $w1x\,2\,1 + w2x\,2\,2 + w3x1x2 + w4x\,2\,3 + w5x1x3 + c = 0$, and a linear hypersurface is defined as $w1x\,2\,1 + w2x\,2\,2 + w3x1x2 + w4x\,2\,3 + w5x1x3 + c = 0$.

In a word, the technique to obtaining a nonlinear SVM is to convert nonlinear data to higher-dimensional linear data as shown in figure. To illustrate how nonlinear translation of original input data into a higher-dimensional space works, take a nonlinear second-order polynomial in a three-dimensional input space.

$$X(x1, x2, x3) = w1x1 + w2x2 + w3x3 + w4x\,2\,1 + w5x1x2 + w6x1x3. \tag{9}$$

The following mappings may be used to translate the three-dimensional input vector $X(x1, x2, x3)$ into the six-dimensional space $Z(z1, z2, z3, z4, z5, z6)$:

$$z1 = \varphi1(x) = x1\,;\, z2 = \varphi\,2(x) = x2\,;\, z3 = \varphi3(x) = x3\,;$$
$$z4 = \varphi4(x) = x\,2\,1\,;\, z5 = \varphi5(x) = x1.x2\,;\, z6 = \varphi6(x) = x1.x3. \tag{10}$$

The modified form of linear data in six-dimensional space will appear as follows. $Z\,w1z1 + w2z2 + w3z3 + w4z4 + w5z5 + w6x1z6 + c$. Thus, if the Z space contains input data for its characteristics $x1$, $x2$, and $x3$ (and thus for Z values), we may use linear decision boundaries to categorise them. The idea of nonlinear mapping and hence of a linear decision boundary appears to be rather straightforward. However, there are several possible complications.

It may be afflicted by the curse of dimensionality, which is frequently linked with large-dimensional data. More precisely, because the number of input instances and support vectors is so vast, it is computationally costly. As a result, the kernel technique is utilised to allocate data points to the hyperplane which is shown in Figure 3.

2.3. Kernel Trick. Cover's theorem is the notion of transforming nonlinearly separable data into linearly separable data. Kernel tactics aid in projecting data points to a higher-dimensional space, where they become more easily separable. Kernel tricks are a technique for computing the dot product of two vectors in order to determine how much they affect one another.

If two input vectors are similar, the dot product can be used as an indicator of how similar they are. The same holds true for the tuple $Xi.Xj$, which serves as an indicator of how similar Xi is to Xj. Since $\varphi(Xi)$ and $\varphi(Xj)$ are the converted features of Xi and Xj in the transformed space, thus, $\varphi(Xi).\varphi(Xj)$ also should be regarded as the similarity measure between $\varphi(Xi)$ and $\varphi(Xj)$ in the transformed space. This is indeed an important revelation and is the basic idea behind the kernel trick

The following conclusions may be drawn from the discussion above.

$$Xi.Xj \Rightarrow \varphi(Xi).\varphi(Xj) \Rightarrow K(Xi, Xj). \tag{11}$$

In the altered space (i.e., the nonlinear similarity measure), this kernel function $K(Xi\,;Xj)$ physically implies the similarity (i.e., nonlinear similarity). The similarity function, K, calculates the degree to which two sets of data, whether originally stored in one set of attributes or after transformation, are similar.

2.4. RBF Kernel as a Projection into Infinite Dimensions. Recall a kernel is any function of the form: $K(Xi, Xj) = \langle\varphi(Xi), \varphi(Xj)\rangle$ where φ is a function that projected vector X into a new vector space. When two projected vectors are sent via the kernel function, it returns the inner product of those two products. Vectors are projected onto an infinitely large space by the function of an RBF kernel. This space is an infinite dimensional Euclidean space for Euclidean vectors. That is, $\varphi_{\text{RBF}} : \mathbb{R}^n \longrightarrow \mathbb{R}^\infty$.

This decreasing function of distance between the axes is how the RBF kernel depicts this similarity. The breadth of the bell-shaped curve is determined by the σ parameter. The bell will become narrower as the value of σ increases. In general, small values of σ produce broad bells.

2.5. Bias-Variance Trade-Off. This is a significant machine learning problem because we want our models to recognise

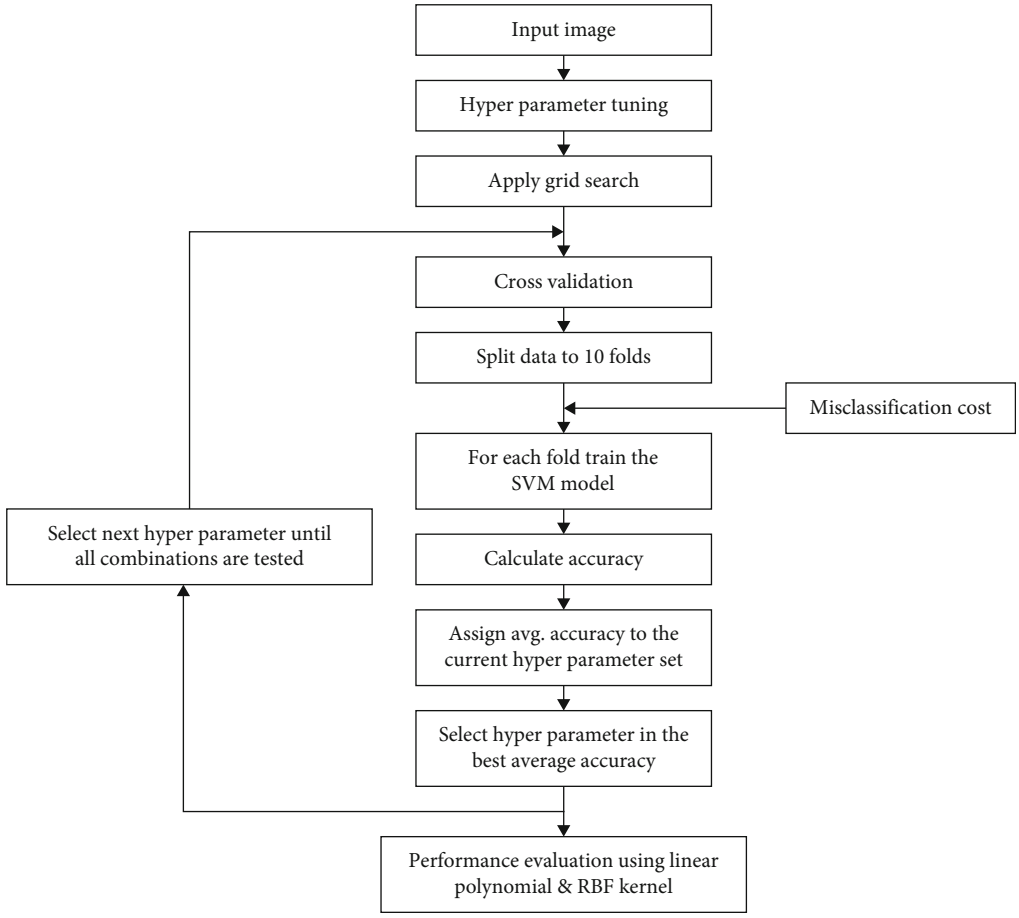

Figure 4: Flow diagram of the proposed method.

specific patterns in the data but not to learn about correlations between the training data's variable x and label y that do not make sense. The bias-variance trade-off is an important issue. An underfitting model does not recognise specific patterns in the training data. On the other side, an overfitting model learns patterns from the training data that are not necessary or even present in the real world. The bias is the difference between the model prediction $f\hat{}(x)$ and the correct output of the underlying function $f(x)$ which we are trying to predict. Because of the restricted number of parameters, simple models often exhibit a large bias when applied to real-world data. On both training and testing data, they frequently have similar error rates.

Bias and variance are important because they aid in finding the best fit for the dataset through hyperparameter adjustment [33]. Tuning aids the model's learning from a given dataset. As a result, the hyperparameter cross-validation approach eliminates overfitting during tweaking.

The deviation of the model prediction $\widehat{f}(x)$ for different training sets is referred to as the variance. This type of model is very sensitive to the training data; hence, it will modify its prediction if it is trained on a new training dataset. As a result, models with a lot of variation tend to do well in training but make a lot of mistakes in testing.

A simple model has a tiny bias, while a more complicated model has a significant variance but little bias. The reason for this is that more advanced models can more closely approximate the goal function (low bias), but the variability of the training set is greatly influenced (leading to high variance). For simple models, the opposite is true. An attempt is made to assess the chance of fraud in a credit card transaction $(y \mid x)$ by using a credit card fraud detection classifier by considering y as the target variable and x as the attributes of transaction. So, we need to be careful and arbitrary when we choose the value of C for better categorisation.

The flow diagram of the proposed method is shown in Figure 4.

2.6. Grid Search and Cross-Validation. Using a technique known as "grid search," hyperparameters can be found that are optimal for a given model. Grid search is extensively used in parameter assortment since it allows us to "brute force" all possible combinations. On a predefined set of criteria, it performs an exhaustive search. The parameter with the greatest score on a criterion is said to be optimum. In [34], the authors found best value for sigma using grid search technique. Grid search generates a model for every possible combination of parameters. It runs through every possible combination of parameters and creates a model for each one. The number of grid divisions determines the number of values that can be stored in each dimension, and we use grid search for this purpose. Uniform sampling

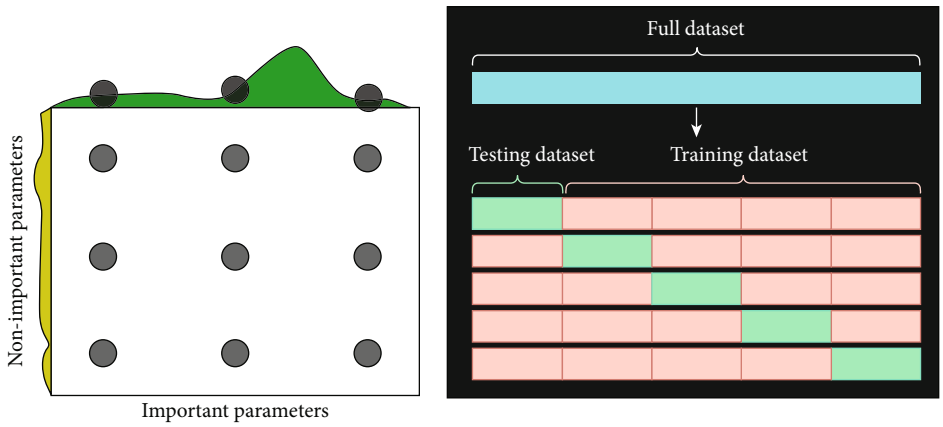

Figure 5: (a) Grid layout. (b) Grid arrangement.

without grid replacement is used to conduct the search in random order. The basic grid arrangement is shown in Figure 5.

Grid search uses the cross-validation (CV) method to optimise the SVM parameters C and σ evaluate performance. Grid search cross-validation is a strategy for selecting the best machine learning model from a set of hyperparameters. Grid search CV searches all grid parameter combinations for a model and delivers the optimal set of parameters with the best performance score. Hyperparameter combinations that can consistently forecast unknown data are what we are looking for. According to [35], cross-validation technique helps to avoid overfitting. We divide the provided data into k sections in order to select C and σ using k-fold CV. Here, we apply five folds; one subset is used for testing, and the other k-1 training subsets are examined as indicated in Figure 4. The predictive model's performance is tested using a number of alternative hyperparameter settings, and the model with the highest cross-validation value is chosen.

2.7. Misclassification Cost. If a bank incorrectly classifies a fraudulent transaction as a nonfraudulent transaction, the bank suffers financial loss. However, if a transaction that is not fraudulent is incorrectly labelled as fraudulent, the bank is just required to provide the customer with a verification notice.

A larger penalty for misclassifications of the minority class can be employed to make up for the imbalance. Therefore, "misclassifying a fraudulent transaction as non-fraudulent is thus more expensive than misclassifying a non-fraudulent transaction as fraudulent."

The SVM tuning approach now includes a new free parameter Cost(i, j), which expresses a cost of categorising a point into class j if its true class is 1 in the form of a numeric square matrix. Accordingly, we penalised those who categorised a sample from class 0 as class 1, while penalising those who classified a sample from class 1 (the minority) as class 0 (the majority). As a result, Cost (0, 1) should be little, whereas Cost (1, 0) should be substantial.

The SVM's decision boundary and ordering are both affected by the misclassification cost, as the hyperplane's orientation varies when the cost function is tweaked. In this way, we can improve the performance of the AUC score.

The essential notion behind the misclassification concept is that if classes are sufficiently represented in the training data, they are treated asymmetrically, with the misclassification cost used to differentiate between non-fraudulent and fraudulent classes. Failure to detect a fraudulent (false negative) has significantly more serious repercussions than misidentifying a nonfraud as fraudulent (false positive). As a result, the cost of misidentifying fraudulent as nonfraudulent will be high, but the cost of misidentifying nonfraudulent as fraudulent will be low. As a result, misclassification costs are used to alter the model's prior class probabilities in order to anticipate fraudulent detection.

3. Experimental Results and Discussions

An online data science site called http://Kaggle.com provided the dataset used in this paper. For the execution, MATLAB 2020b was used, and the processor is an Intel Core i5-3470 CPU @ 5.32 GHz, 8 GB RAM, 500 GB HDD and a 64-bit Windows 10 operating system. The time stamp is the first field, and the monetized card transaction amount is the final field. Each data entry has 30 fields. Only 492 or 0.17 percent of the 284,807 credit card transactions in the database are fraudulent. Most transactions were less than 100 INR, but some might go up to 10,000 INR.

This is how we are going to approach the classification problem:

(i) The first process is to apply SVM to the data and to use the results as benchmark

(ii) The optimized SVM using grid search cross-validation and misclassification cost is applied to the data

(iii) SVM outcomes will be analysed utilising linear, polynomial, and RBF kernel functions

It is common practise to utilise the confusion matrix to show how a machine learning classifier's prediction does not match the dataset's ground truth. The following are listed in the matrix of confusion:

(i) True positive (TP)

It is the proportion of positive labels predicted accurately by trained models. This is the number of samples classified as class 1 that were accurately anticipated to be class 1 (fraudulent).

(ii) True negative (TN)

It is the number of incorrectly predicted negative labels by trained models. This is the number of samples classified as class 0 that were accurately predicted to be class 0.

(iii) False positive (FP)

It is the number of mistakenly predicted positive labels by trained models. The number of class 1 samples was forecasted wrongly as class 0.

(iv) False negative (FN)

It is the number of mistakenly predicted negative labels by trained models. This is the number of class 0 samples that were forecasted as class 1 wrongly.

To have a perfect model, all of the cases would be projected as being positive and none as being negative, resulting in the null values of FN and FP. Classifier performance can be assessed using a variety of measures derived from the confusion matrix.

(i) Recall

Recall is a measure that indicates how accurately our model recognises true positives.

$$\text{Recall} = \frac{\text{true positives}}{(\text{true positives} + \text{false negatives})}. \tag{12}$$

(ii) Precision

It is defined as the ratio of true positives to all positives.

$$\text{Precision} = \frac{\text{true positives}}{(\text{true positives} + \text{false positives})}. \tag{13}$$

(iii) $F1$-score

This score represents the harmonic mean of precision (P) and recall (R).

$$F1 - \text{score} = \frac{2 * P * R}{(P + R)}. \tag{14}$$

(iv) Error rate

It is considered by the no. of predictions that were wrong is divided by the total number of observations. When it comes to error rates, 0.0 is perfect, whereas 1 is optimal.

$$\begin{aligned}\text{ERR} &= \frac{\text{false positives} + \text{false negatives}}{\text{true positives} + \text{true negatives} + \text{false negatives} + \text{false positives}} \\ &= \frac{\text{false positives} + \text{false negatives}}{P + N}.\end{aligned} \tag{15}$$

(v) Accuracy

When determining the accuracy of a prediction, divide the total no. of correct predictions by entire no. of observations contained in a dataset (ACC). At the most precise level, 1.0, and at the least precise level, 0.0. Additionally, it can be written as 1–ERR.

$$\begin{aligned}\text{ACC} &= \frac{\text{true positives} + \text{true negative}}{\text{true positives} + \text{true negative} + \text{false negatives} + \text{false positives}} \\ &= \frac{\text{true Positives} + \text{true negative}}{P + N}.\end{aligned} \tag{16}$$

Table 1 shows that a variety of kernels have been used to study the detection of credit card fraud. The cost of misclassification must also be calculated along with these kernel parameters. In order to compensate for the imbalance, the misclassification cost is utilised to impose a greater penalty on misclassifications of minorities. For all kernel functions, a grid search is used to determine the best parameter values.

Table 1 lists all of the confusion matrix measurements, as well as the overall cost of misclassification, which is computed by adding the cost matrix by the confusion matrix established by the indicated technique. When the total cost of misclassifying is low, it makes sense to penalise people who change their prior probabilities during hyperparameter tweaking for better fraud detection.

Hence, the reported sensing technique shows that SVM optimised with RBF and misclassification cost performs well in fraudulent credit card detection better than other approaches.

To determine the method's false discovery rates, the positive predictive values are plotted against the false discovery rates (FDRs). PPV describes the likelihood that a favorable outcome is correct. The false discovery rate in statistics is defined as the ratio of mistakenly positive outcomes to all positive results. For identifying credit card fraud, the percentage values of the elements below the diagonal of the confusion matrix are insignificant. These data illustrate examples of how a customer's credit rating is computed following the proposed technique. In fact, the rate of false positives is dropping. Model 3 (SVM optimised with RBF and misclassification cost) surpasses the others in terms of credit card fraud prediction, as shown in Figure 6.

It is prudent to begin by analysing classifier performance using a confusion matrix-based score. It is possible that some consumers want to know more about how the

Table 1: Confusion matrix measures and the overall misclassification cost.

Approach	Recall	Precision	$F1$-score	Error rate	Accuracy	Misclassification cost
SVM	0.9492	0.9878	0.9681	0.0325	0.9674	NA
Linear optimized SVM	0.9603	0.9837	0.9718	0.0284	0.9715	52.0
RBF optimized SVM	0.9739	0.9878	0.9808	0.0193	0.9836	34.6
Quadratic optimized SVM	0.9603	0.9837	0.9718	0.0284	0.9715	52.0
Cubic optimized SVM	0.9660	0.9837	0.9747	0.0254	0.9745	45.5

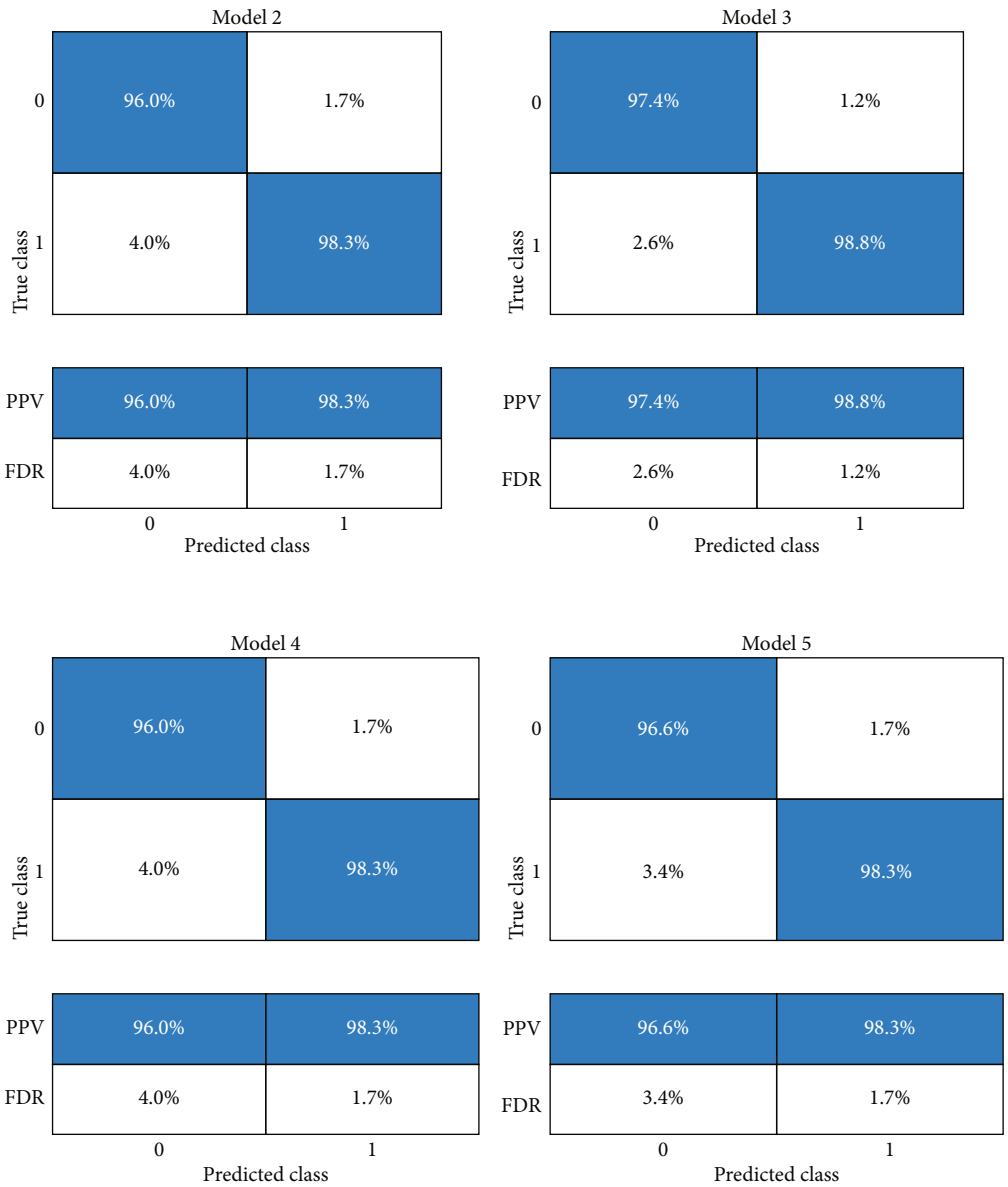

Figure 6: False discovery rates.

classifiers perform over the entire testing set. Metric assessment is used in order to get a clearer picture of the classifier's behaviour. Receiving operating characteristic (ROC) curves are used a lot in parametric evaluation to see how well a new method works.

One measure to gauge the accuracy of a test is the number of false positive results. The true positive rate vs. the false positive rate is shown in Figure 5. A positive genuine positive rate is shown, while a negative false positive rate is shown on the x- and y-axes, respectively. It is advantageous

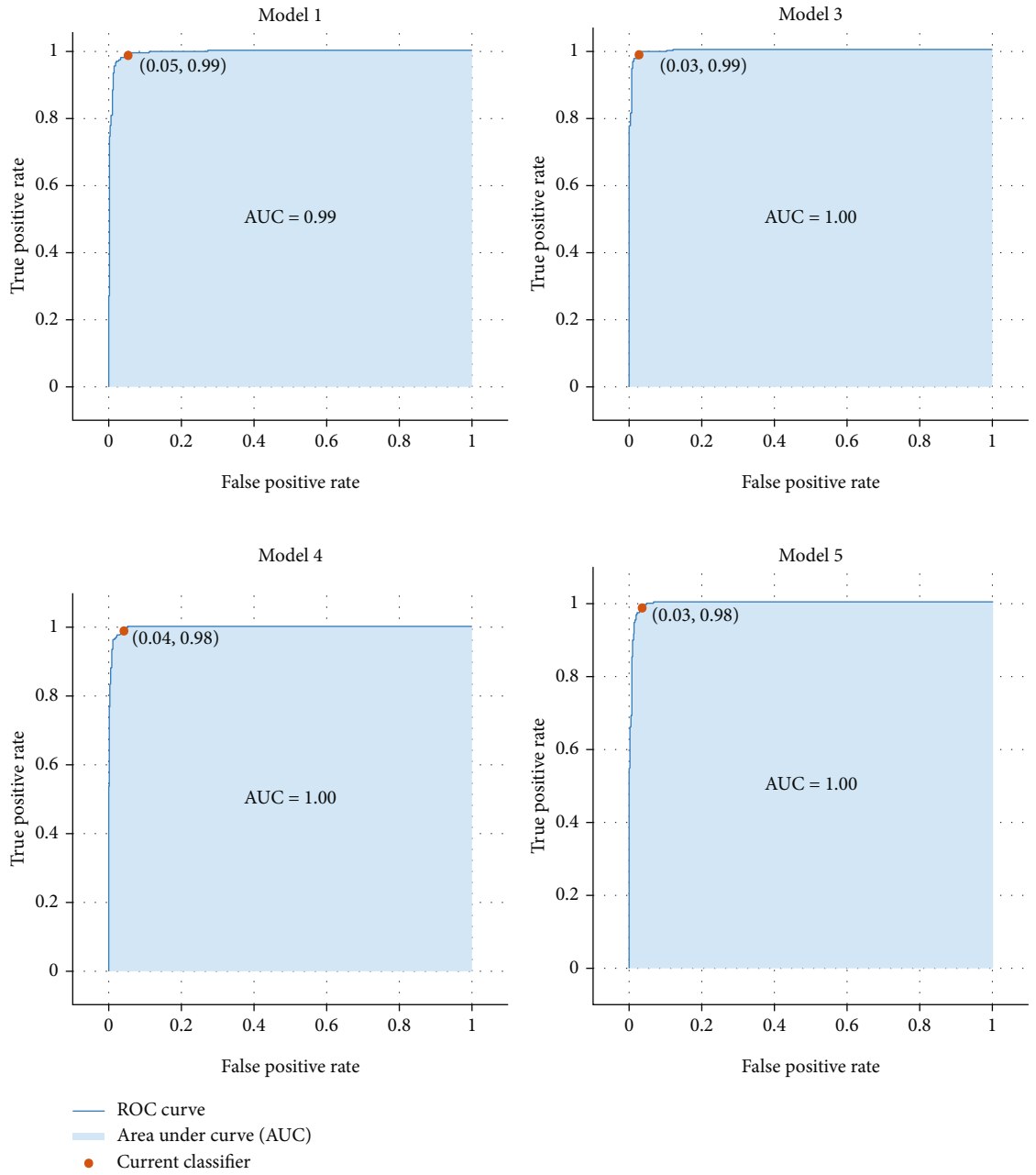

Figure 7: AUC-ROC of our models.

to use a ROC curve to evaluate a classifier rather than misclassification error since ROC plots may be generated for all possible thresholds between 0 and 1. It shows the receiver operating characteristic curves of models 1, 3, 4, and 5. Model 3 outperforms all others when the x- and y-axes are set to a value of 0.03 and 0.99, respectively.

There are some of additional options available as well. The following is a list of the most often encountered:

False negative rate: it is the possibility that a test will miss a real positive, known as the false negative rate, or miss rate. As a result, we have FN/FN + TP, where FN is the number of false negatives and TP is the number of true positives.

The probability of fraudulent is known as the positive predictive value. The formula is TP/TP + FP = TP/TP + FP.

To measure a classifier's performance, the ROC curve might offer a higher score to the classifier, known as the area under the curve (AUC). It is feasible to set the threshold that corresponds to the spot on the ROC curve that best depicts the trade-off between sensitivity and specificity. To evaluate the model without regard to the setting of a threshold, ROC curves can be used. Models with a high AUC are referred to as having a high level of expertise. AUC values for models 3–5 are shown in Figure 7. It is still close to one, though, with a true positive rate of 99.9% for model 3. As a result, when compared to other models, ours has a high AUC score.

The proposed approach is compared with some of the machine learning algorithm, and it is shown in Table 2.

Table 2: Performance between various machine learning algorithms.

Model	Accuracy	Recall	Precision	F1-score
Random forest	83.78%	79.64%	92.78%	85.71%
Logistic regression	79.91%	59.29%	81.70%	68.71%
Decision tree	89.91%	79.64%	68.70%	73.77%
Naïve Bayes	78.14%	83.18%	6.73%	12.46%
SVM-RBF	98.36	97.39	98.78	98.08

Table 3: Performance between our models with other recent models.

Approach	Precision	Accuracy
CS-SVM	98	98
Weighted SVM	93.7	97
Proposed (RBF)	98.78	98.36

Table 2 shows that SVM with RBF gives better performance than other methods.

Weighted support vector machines [36] and cuckoo search SVM (CS-SVM) [3] are used to compare the results of the proposed sensing machine learning experiments. CS-SVM is excellent for continuous issues, but it also offers flexibility and appropriate search restrictions for discrete problems. If the appropriate weight is not assigned to each individual data point in weighted SVM, accuracy suffers. However, our suggested method addresses these two difficulties by combining grid search cross-validation with misclassification cost. According to the results shown in the table, the RBF-based approach is more precise and accurate than other approaches as shown in Table 3.

4. Conclusion

Categorisation algorithms are evaluated using a new sensing machine learning method that takes the cost of incorrect classification into account. Grid search cross-validation optimization of the SVM's hyperparameters and the theory of replicating kernels such as linear, Gaussian, and polynomial simultaneously increase the model's robustness. The findings show that this method significantly improves the finding of credit card fraud.

The quality of detection can be still improved. If the following characteristics are taken into account, ample data, high-quality data, and elements that are data should all be well-structured and free of bias.

However, the suggested technique has a significant downside in terms of quick convergence and complexity. Grid search's complexity is projected to develop exponentially at a rate of $O(n^k)$ if k parameters with n different values are examined. This may be overcome by examining a larger dataset and doing a random search.

Although SVMs are not ideal for managing huge datasets, they perform poorly when noise is present in the data (e.g., overlapping classes). Future studies might look at the sequence of fraud and genuine transactions before credit cards are revoked. Future studies may also look at the distinctions between other forms of fraud, such as the differences in behaviour between stolen and counterfeit cards. It might also include noise which can be removed by autoencoder.

Data Availability

The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

There are no known financial or personal conflicts of interest that would have impacted the research presented here.

References

[1] D. Wang, B. Chen, and J. Chen, "Credit card fraud detection strategies with consumer incentives," *Omega*, vol. 88, pp. 179–195, 2019.

[2] N. Rtayli and N. Enneya, "Enhanced credit card fraud detection based on SVM-recursive feature elimination and hyper-parameters optimization," *Journal of Information Security and Applications*, vol. 55, article 102596, 2020.

[3] C. Li, N. Ding, H. Dong, and Y. Zhai, "Application of credit card fraud detection based on CS-SVM," *International Journal of Machine Learning and Computing*, vol. 11, no. 1, pp. 34–39, 2021.

[4] 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, Istanbul, Turkey, 2011.

[5] C. Liu, Y. Chan, K. Alam, H. Syed, and H. Fu, "Financial fraud detection model: based on random forest," *International Journal of Economics and Finance*, vol. 7, no. 7, pp. 178–188, 2015.

[6] L. Mukhanov, "Using Bayesian belief networks for credit card fraud detection," in *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications*, pp. 221–225, Insbruck, Austria, 2008.

[7] S. V. S. S. Lakshmi and S. Deepthi Kavila, "Machine learning for credit card fraud detection system," *International Journal of Applied Engineering Research*, vol. 13, no. 24, pp. 16819–16824, 2018.

[8] E. Ileberi, Y. Sun, and Z. Wang, "A machine learning based credit card fraud detection using the GA algorithm for feature selection," *Journal of Big Data*, vol. 9, no. 1, p. 24, 2022.

[9] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, "Random forest for credit card fraud detection," in *2018 IEEE 15th international conference on networking, sensing and control (ICNSC)*, pp. 1–6, Zhuhai, China, 2018.

[10] A. Syarif, G. Prugel-Bennett, and G. Wills, "SVM parameter optimization using grid search and genetic algorithm to improve classification performance," *Telkomnika*, vol. 14, no. 4, pp. 1502–1509, 2016.

[11] C. J. Fu and Y. P. Yang, "A batch-mode active learning SVM method based on semi-supervised clustering," *Intelligent Data Analysis*, vol. 19, no. 2, pp. 345–358, 2015.

[12] K. Kianmehr and R. Alhajj, "Effectiveness of support vector machine for crime hot-spots prediction," *Applied Artificial Intelligence*, vol. 22, no. 5, pp. 433–458, 2008.

[13] V. Vijayaraghavan and M. Laavanya, "Vehicle classification and detection using deep learning," *International Journal of Engineering and Advanced Technology*, vol. 9, no. 1S5, pp. 24–28, 2019.

[14] M. Laavanya and V. Vijayaraghavan, "Real time fake currency note detection using deep learning," *International Journal of Engineering and Advanced Technology*, vol. 9, no. 1S5, pp. 95–98, 2019.

[15] M. Laavanya and V. Vijayaraghavan, "Image denoising with a convolution neural network using Gaussian filtered residuals," *IEIE Transactions on Smart Processing and Computing*, vol. 10, no. 2, pp. 96–100, 2021.

[16] M. Zanin, M. Romance, S. Moral, and R. Criado, "Credit card fraud detection through parenclitic network analysis," *Complexity*, vol. 2018, Article ID 5764370, 9 pages, 2018.

[17] H. Tingfei, C. Guangquan, and H. Kuihua, "Using variational auto encoding in credit card fraud detection," *IEEE Access*, vol. 8, pp. 149841–149853, 2020.

[18] C. Ying and Z. Ruirui, "An overview of computational models for industrial Internet of Things to enhance usability," *Complexity*, vol. 2021, Article ID 6618841, 11 pages, 2021.

[19] I. Benchaji, S. Douzi, B. El Ouahidi, and J. Jaafari, "Enhanced credit card fraud detection based on attention mechanism and LSTM deep model," *Journal of Big Data*, vol. 8, no. 1, 2021.

[20] M. Zareapoor and P. Shamsolmoali, "Application of credit card fraud detection: based on bagging ensemble classifier," *Procedia Computer Science*, vol. 48, pp. 679–685, 2015.

[21] M. Laavanya and M. Karthikeyan, "Dual tree complex wavelet transform incorporating SVD and bilateral filter for image denoising," *International Journal of Biomedical Engineering and Technology*, vol. 26, no. 3/4, pp. 266–278, 2018.

[22] V. Vijayaraghavan and M. Karthikeyan, "Denoising of images using principal component analysis and undecimated dual tree complex wavelet transform," *International Journal of Biomedical Engineering and Technology*, vol. 26, no. 3/4, pp. 304–315, 2018.

[23] S. Caner and D. Fabio, "The impact of different kernel functions on the performance of scintillation detection based on support vector machines," *Sensors*, vol. 19, no. 23, p. 5219, 2019.

[24] B. James and B. Yoshua, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[25] E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter tuning for machine learning algorithms used for Arabic sentiment analysis," *Informatics*, vol. 8, no. 4, p. 79, 2021.

[26] I. Benchaji, S. Douzi, and B. El Ouahidi, "Credit card fraud detection model based on LSTM recurrent neural networks," *Journal of Advances in Information Technology*, vol. 12, pp. 113–118, 2021.

[27] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: a big comparison for NAS," *Computer Science*, vol. 1912, pp. 1–11, 2019.

[28] S. Yuan and F. Chu, "Support vector machines-based fault diagnosis for turbo pump rotor," *Mechanical Systems and Signal Processing*, vol. 20, no. 4, pp. 939–952, 2006.

[29] J. Qu, Z. Liu, M. J. Zuo, and H.-Z. Huang, "Feature selection for damage degree classification of planetary gearboxes using support vector machine," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 225, no. 9, pp. 2250–2264, 2011.

[30] A. Villa, M. Fauvel, J. Chanussot, P. Gamba, and J. A. Benediktsson, "Gradient optimization for multiple kernel's parameters in support vector machines classification," in *Proceedings of IEEE International Conference on Geoscience and Remote Sensing Symposium*, Boston, 2008.

[31] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, second edition, 2009.

[32] N. Christianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, UK, 2013.

[33] P. Mehta, M. Bukov, C.-H. Wang et al., "A high-bias, low-variance introduction to machine learning for physicists," *Physics Reports*, vol. 810, pp. 1–124, 2019.

[34] E. Y. Widodoa, J.-D. S. Kimb, B.-S. Yang et al., "Fault diagnosis of low speed bearing based on relevance vector machine and support vector machine," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7252–7261, 2009.

[35] S. W. Lin, K. C. Ying, S. C. Chen, and Z. J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[36] D. F. Zhang, B. Bhandari, and D. Black, "Credit card fraud detection using weighted support vector machine," *Applied Mathematics*, vol. 11, no. 12, pp. 1275–1291, 2020.