Operational Research (2024) 24:57
https://doi.org/10.1007/s12351-024-00864-3

ORIGINAL PAPER

A hybrid metaheuristic optimised ensemble classifier with self organizing map clustering for credit scoring

Indu Singh[1] · D. P. Kothari[2] · S. Aditya[1] · Mihir Rajora[1] · Charu Agarwal[3] · Vibhor Gautam[4]

Received: 23 June 2023 / Revised: 18 August 2024 / Accepted: 23 August 2024

Abstract
Credit scoring is a mathematical and statistical tool that aids financial institutions in deciding suitable candidates for the issuance of loans, based on the analysis of the borrower's financial history. Distinct groups of borrowers have unique characteristics that must be identified and trained on to increase the accuracy of classification models for all credit borrowers that financial institutions serve. Numerous studies have shown that models based on diverse base-classifier models outperform other statistical and AI-based techniques for related classification problems. This paper proposes a novel multi-layer clustering and soft-voting-based ensemble classification model, aptly named Self Organizing Map Clustering with Metaheuristic Voting Ensembles (SCMVE) which uses a self-organizing map for clustering the data into distinct clusters with their unique characteristics and then trains a sailfish optimizer powered ensemble of SVM-KNN base classifiers for classification of each distinct identified cluster. We train and evaluate our model on the standard public credit scoring datasets—namely the German, Australian and Taiwan datasets and use multiple evaluation scores such as precision, F1 score, recall to compare the results of our model with other prominent works in the field. On evaluation, SCMVE shows outstanding results (95% accuracy on standard datasets) when compared with popular works in the field of credit scoring.

Keywords Machine learning · Credit scoring · Self organizing maps · Ensemble classification · Sailfish optimizer

1 Introduction

Lending money is one of the primary businesses of financial institutions as it reaps huge profits for its stakeholders. Although, market fiascos of the likes of credit rationing and risk may occur due to the absence of the applicant's

Extended author information available on the last page of the article

Published online: 18 September 2024

information. Credit risk is a major threat to financial institutions. It can exhibit an exorbitant effect on the financial market, especially during a financial crisis period. The past financial crises have damaged many countries financially, the effects of which will long be felt in the future. Developing efficient risk management has thus become a major focus for financial institutions. Banks have been increasingly using statistical and ML models for credit score evaluation. While huge advancements have already been made in the field, according to a survey conducted by Dastile et al. (2020), bank regulators stand with the belief that models should be transparent. Despite the non-transparency of deep learning models, they have still been gaining prominence. A justification to overcome the problem of transparency is to rationalize the deep learning models, i.e., be able to find reasons as to why the model generates a particular decision.

Credit scoring is a robust mathematical and statistical model which helps in mitigating the problem of credit defaults and managing credit risks by computing the credit risk associated with a customer. The usage of credit scoring increases the accessibility of financial institutions to a larger demographic, especially in developing countries where greater financial accessibility is a must for the development of the country as a whole, as noted in a study by Bumacov et al. (2014). Credit scoring has also been shown to increase the efficiency of loan officers, leading to a causal effect that leads to the handing out of more loans and serving more borrowers. The rationale behind credit scoring is to use customer's past data such as credit history, income, and marital status and calculate their ability to repay the loan. There are two primary reasons to develop a powerful and robust credit scoring model. Firstly, considering the massive volume of the credit market, even a minute upgradation in the classification accuracy are able to reap a substantial amount of profit. Moreover, regulations regarding credit risk management are getting strict.

The Basel Community on Banking Supervision(BCBS) monitors and guides financial institutions to ensure credit security. Any violation of the guidelines can cause enormous regulation costs. According to a study by Reichert et al. (1983), the previously used statistical models are not absolutely efficient in mapping out real world scenarios. They do, however, conclude that these models' inability to meet the assumptions of fair credit scoring models is less significant as opposed to the ability of credit evaluation officers' comprehension of the classification model and its shortcomings.

Credit scoring as a comprehensive procedure can be subdivided into 4 different stages—application scoring, scoring on basis of behaviour, collection scoring, and fraudulent activity detection. In this study, our major focus lies on predicting if an applicant is a 'risky borrower' which equates to a binary classification problem.

In the initial stages of credit scoring, statistical techniques such as logistic regression and linear discriminant analysis(LDA) grew popular. Statistical techniques are widely used because of their easy implementation. However, these are based on some inferences such as linear separation and normal distribution of data, assumptions which are generally violated in the real world. Neural network models for credit scoring were also seen in the early 2000's, as were noticed in a publication by West (2000). With the rapid development of artificial intelligence, techniques

making extensive use of machine learning algorithms such as KNN, ANN (Artificial Neural Network), SVM, artificial immune systems have been introduced.

In most of the studies, single classifier-based models are constructed for credit scoring. Experiments have shown that singular classifiers are ineffective in capturing the patterns of individual customers, and thus a different type of classification model, ensemble classifiers have been employed that have proved effective for improving the accuracy and stability of single classifiers. Ensemble-based classifiers combine several base classifiers for better predictive performance (Singh 2017). Thus, research on ensemble methods has grown popular in the last few years. The main idea behind ensemble classification is to overcome the weakness of individual classifiers by combining their results. For effective ensemble classification, the base classifiers must be accurate, diverse, and sensitive. The construction of ensemble models usually consists of three steps—pool generation, selection of the classification and related models, and the combination of the delivered results. In the first step, base models are generated to create a model pool using various classification algorithms or data. The second step is necessary to enhance the model performance. The pool models are selected based on certain rules to maintain an accurate model. In the last step, the selected models are combined based on some heuristics and rules.

This paper proposes a multi-layer clustering and classification model consisting of primary classifiers, using SOM and metaheuristic algorithm. In the first step, instead of a traditional artificial neural network algorithm, we use self-organizing maps (SOM) for clustering our dataset. SOM models are unsupervised ANN-based clustering techniques based on competitive learning in which the nodes associated with weights compete with each other to win an input pattern. It is used to represent the high-dimensional input space to the final low-dimensional space. Once our dataset is divided into different clusters where each data point is now only associated with data points within the same cluster, we apply a classification model to each cluster individually. We use an ensemble classifier and metaheuristic algorithm for increasing the overall efficiency of our model. Our ensemble model is formed using 6 sets of KNN and 6 sets of SVM. The classifiers used have different prediction capabilities. We use SVM for its proven results in classifying non-linearly separable data and KNN for its closest distance-based classification.

Application of metaheuristics to classification problems has been observed in the literature as early as 2007. For instance, Ant Colony Optimization was used in standard classification problems (Martens et al. 2007). In our approach, the latest metaheuristic model, sailfish optimizer (SFO) is employed for weights generation for our ensemble model. The relative performance of the classifier is considered where the classifier that performs well on the cluster dataset is assigned higher weights. Based on the behavior of sailfish hunting for sardines, the optimizer has shown excellent results when tested against various optimizers. SFO provides advantages in terms of its exploration and exploitation phases, a formidable convergence speed in reaching global optimization, and also better efficiency.

In all previous research, the model is trained using the entire dataset. Our model departs from existing models by initially clustering similar data points and applying the classification model to each of them. Also, the use of diverse base classifiers

in our ensemble model yields excellent results. Another advantage of our proposed model is the use of SOMs to find better sets of clusters of the dataset. This training technique develops highly efficient state-of-the-art models that can provide high efficiency.

Thus, SCMVE offers the following advantages and novelties compared to existing work in the same field:

- *Clustering through Self Organising Maps that identifies groups with distinct characteristics that can be used to identify and train the classification models on*
- *Use of diverse base classifiers in soft-voting ensemble model, an approach that outperforms singular complex models such as artificial neural networks*
- *Utilising an optimized metaheuristic sailfish optimizer for weights optimization of the ensemble model, leading to enhanced performance on various classification metric evaluations*
- *Application of a unique training strategy, where local fitting of classifier models are done to the formed clusters (as opposed to training on the whole dataset), hence utilising the distinct characteristics of each cluster found through SOM clustering phase leading to outstanding classification results.*

The remainder of this paper is structured as follows: Sect. 2 provides a summary of previous work done in credit scoring. Section 3 explains the methodology of SCMVE. Section 4 introduces the experimental design, followed by the experimental results in Sect. 5. The conclusion and future work are provided in Sect. 6.

2 Related work

Since credit scoring is fundamentally a binary classification problem, there exist models that employ the use of a single classifier to predict a credit defaulter.

Earlier models mostly relied on existing statistical methods such as Linear Discriminant Analysis (as proposed by Fisher 1936) or the usage of Logistic Regression and Superscorecards as proposed by Hand and Kelly (2002) These basic models performed daily however failed to provide reliable results given unrepresentative or new samples.

However, with the advent of machine learning techniques the aim was set to solve the issues associated with the statistical methods. Models based on k-Nearest-Neighbours (Henley and Hand 1996), Support Vector Machines (Li et al. 2006), Classification Trees (Li et al. 2004) and Neural Networks (West 2000) were employed to considerable success for solving the same.

2.1 Neural network models

Lee et al. (2002) proposed a hybrid neural discriminant model which aimed to simplify the neural networks by incorporation of the traditional discriminant analysis. They employed LDA to first model the credit scoring problem and identify

the significant input variables, which are then passed on to the Artificial Neural Network to perform the prediction. Hence, the LDA acted as a tool to design the topology of the network.

Onan (2019b) proposed a two-stage neural network framework for topic extraction, using advanced embeddings (word2vec, POS2vec, word-position2vec, LDA2vec) and an ensemble clustering approach.

Lappas and Yannacopoulos (2021) are able to combine expert knowledge and genetic algorithms into a machine learning approach to supplement supervised credit scoring with expert input, increasing the overall efficiency. Safi et al. (2022) combine a neural network with metaheuristic optimizers and five cost sensitivity fitness functions as the base learners in a majority voting ensemble learning paradigm.

Onan (2023a) proposed a novel GTR-GA approach, which integrates graph-based neural networks with genetic algorithms for text augmentation, effectively enhances data diversity and improves model performance.

2.2 Self organising map models

As an extension to the artificial neural network models used for the classification step, self organising maps, which are a widely employed neural network model, have been used instead of the traditional ANN models. Unlike the ANN, the SOM models are unsupervised and produce a topology preserving mapping between the input parameter set (which is a higher dimensional map space) to the final credit score (a lower dimensional map space). Lau (2006) have explored different SOM Models that can be employed for classification problems.

Hsieh (2005) proposed a hybrid mining approach which used a self organising map for deciding the parameters of the K Means clustering algorithm followed by an Artificial Neural Network for classifying the samples. Suleiman et al. (2021) also used an SOM to improve the ability of pattern recognition in the data and utilise that to increase the efficiency of K-Means Classifier and neural networks.

AghaeiRad et al. (2017) developed a hybrid model for credit scoring using SOM clusters and feedforward neural networks. They used the knowledge obtained from SOM clusters and passed it as information to be trained on by the neural network classifier. This gave better results compared to a standalone FNN due to the increased information present in the input.

Onan (2019a) proposed a consensus clustering-based undersampling method to tackle class imbalance in machine learning. By undersampling the majority class with various clustering algorithms and evaluating performance using different supervised and ensemble learning methods.

Onan (2021b) introduces sentiment analysis on product reviews based on weighted word embeddings and deep neural networks. The architecture includes a weighted embedding layer, convolutional layers (1-g, 2-g, 3-g), max-pooling, LSTM, and a dense layer.

Onan (2022) proposed bidirectional convolutional recurrent neural network employs bidirectional LSTM and GRU layers to capture both past and future contexts.

2.3 Metaheuristic algorithms in credit scoring

Genetic algorithms have been used widely and in different capacities in the credit scoring problem.

Onan et al. (2016b) employed a multiobjective differential evolution algorithm to optimize classifier and class weights within a static classifier selection framework for sentiment analysis.

Onan (2018a) optimized the Latent Dirichlet Allocation (LDA) parameters by employing a swarm intelligence approach. They utilized metaheuristic algorithms like Particle Swarm Optimization (PSO) to estimate the number of topics and other key parameters in LDA. This complements He et al. (2018) work on adaptive models.

Pławiak et al. (2019) published a work constituting an application of genetic cascading of ensemble classifiers. They combined the benefits of evolutionary algorithms and ensemble classifiers, alongside using deep learning to develop a complex credit scoring model. They used genetic algorithms in three different instances: first for feature selection on the Australian dataset, second for hyperparameter optimization, and a third for deriving a training technique for selecting the classifiers for the final trained model.

Kozodoi and Lessmann (2020) improved on the feature selection through GA by modelling it into a multiobjective task. They then used a Particle Swarm Optimization(PSO) algorithm and evaluated it using three different fitness functions based on: the number of features, the relative acquisition costs of the features and the AUC-ROC curve fitness score of the trained model.

Tripathi et al. (2020) proposed a novel Binary BAT algorithm on the credit scoring dataset. They combined it with a radial Neural Network (RBFN) for a hybrid credit scoring model. They used the metaheuristic algorithm for feature selection on the dataset and used the Neural Network for training the selected features in a supervised classification task.

Simumba et al. (2022) incorporated stakeholder requirements into their model during feature selection and used that to train their supervised classification model. They compared the results of two modified metaheuristic algorithms: a Grasshopper algorithm integrated with non-dominated sorting and genetic algorithm, and a genetic algorithm integrating different selection, crossover, and mutation strategies. They evaluated their results with empirical data collected from farmers in Cambodia.

Şen et al. (2020) proposed multilevel metaheuristic algorithms for credit scoring. They used an SVM classifier, combined with a Genetic Algorithm in a two-level feeding mechanism for increased model accuracy. They used the GA first to find the optimized parameters of the SVM model, and then used it for feature selection on the dataset that increases the classification accuracy.

Onan (2023b) proposed SRL-ACO, a text augmentation framework that uses Semantic Role Labeling (SRL) and Ant Colony Optimization (ACO) to generate additional training data for NLP models. SRL-ACO enhances data quality by preserving semantic roles in new sentences. Experimental results demonstrate improved performance in sentiment analysis, toxic text detection, and sarcasm identification tasks.

2.4 Ensemble models in credit scoring

Ensemble models and models based on genetic algorithms combine the power of machine learning models to generate highly accurate credit risk assessment models and assist us in classifying previously unrepresentative samples in the dataset.

Van Gestel et al. (2003) proposed a Least Squares Support Vector Machine (LS-SVM) classifiers within the Bayesian evidence framework to predict the tendency of an entity to default on their credit given a set of parameters. Once the probabilities for the defaulting of an entity are generated a sensitivity analysis with respect to the input set is carried out which provides us with an insight into the parameters that are affecting the creditworthiness of the entity.

Onan et al. (2017) work on hybrid ensemble pruning enhances classifier diversity and performance, aligning with the approach of improving sentiment classification accuracy through advanced ensemble methods.

Onan (2021a) work on hybrid ensemble pruning improves classifier diversity and performance, enhancing sentiment classification accuracy through advanced ensemble methods.

Onan (2018b) integrated a Random Subspace ensemble of Random Forest classifiers with four types of features-authorship attribution, character n-grams, part of speech n-grams, and discriminative word frequency.

Hsieh and Hung (2010) investigated the approach involving the proper preprocessing of the dataset into homogenised clusters followed by the classification of the samples into the preprocessed categories. The ensemble model hence proposed resulted in an efficient ensemble classifier.

Onan et al. (2016a) combined five statistical keyword extraction methods with various ensemble techniques for text classification, including Naïve Bayes, SVM, logistic regression, and Random Forest, akin to how Zhang et al. (2021) utilized ensemble techniques for improving credit risk models. This integration of feature

extraction with ensemble methods aligns with the advancements in robust predictive modeling.

He et al. (2018) improved on the construction of the ensemble models by basing their adaptability to different imbalance ratios by the supervised undersampling of the dataset (based on the estimation of the data imbalance ratio), followed by classification by tree-based base classifiers which classify samples in the respective data subsets. Finally, a particle swarm optimization algorithm was applied to the base classifiers to obtain the final ensemble model.

Zhang et al. (2021) were able to develop a novel ensemble model that used a local outlier factoring algorithm added with a bagging strategy to construct a trained model that works on the outlier adaptability of base classifiers. They combined novel methods of feature reduction and ensemble learning methods for parameter optimization and finally used a stacking based ensemble model to train the dataset on.

Nalič et al. (2020) used ensemble techniques for feature selection on datasets and proposed the if_any voting method that was able to outperform other standard voting procedures. They combined linear models, SVMs, naive Bayes and decision tree classifiers into a soft voting ensemble model.

Xia et al. (2020) proposed a new tree-based overfitting-cautious heterogeneous ensemble model for credit scoring. The suggested method could dynamically give weights to base models based on the overfitting metric during ensemble selection.

To improve the prediction performance of credit scoring, Tripathi et al. (2019) created a hybrid model that combines feature selection and a multilayer ensemble classifier architecture. The first phase is preprocessing, which sets ranks and weights of classifiers, followed by the ensemble feature selection approach, and finally, the dataset with the selected features is used in a multilayer ensemble classifier architecture. In addition, since classifier placement influences the ensemble framework's predictive performance, a classifier placement algorithm based on the Choquet integral value was devised.

Xia et al. (2018) introduced a new heterogeneous ensemble credit model that combined the bagging and stacking algorithms. In three ways, the proposed model varied from the existing ensemble credit models: pool creation, base learner selection, and trainable fuser.

To increase the prediction performance, Guo et al. (2019) proposed a novel multistage self-adaptive classifier ensemble model based on statistical approaches and machine learning techniques. First, the original data was processed into a standardized, representative sample using a multistep data preparation method. Second, based on their performance, base classifiers were self-adaptively picked from the candidate classifier repository, and their parameters were adjusted using the Bayesian optimization algorithm post which the ensemble model was integrated using these optimized base classifiers, and used multilayer stacking to generate new features and particle swarm optimization to achieve the classifier weights in the ensemble model.

Xiao et al. (2020) devised a GMDH-based cost-sensitive semi-supervised selective ensemble (GCSSE) model using a semisupervised, cost-sensitive learning, group method of data handling (GMDH), and an ensemble learning strategy.

Liu et al. (2022) proposed a multigrained and multi-layered gradient boosting decision tree (GBDT) that effectively increases the diversity of prediction and further improves the performance of credit scoring by providing more precise credit scoring results. The classification techniques for credit scoring have been summarised in Table 1.

3 Proposed methodology

This section gives a detailed description of SCMVE which is the framework used to determine if a data point belongs to good credit or risky credit. SCMVE uses the concept of multi-layer clustering using an artificial neural network-based technique (Self Organising Map) and then uses a soft voting-based ensemble classifier to predict the final class of the test data point.

The weights for our ensemble model are generated using a metaheuristic optimization technique—Sailfish Optimizer.

The overall architecture of SCMVE is divided into 2 phases:

- **Clustering Phase**: The entire dataset is clustered with data in them having similar feature values. The idea of cluster is based on the fact that different set of controlling attributes which affect the final classification are better identified and trained accordingly when dealt with as a cluster.
- **Classification Phase**: Once data is divided into cluster regions, the classification models predict the final class of data points. To achieve classification, we use an ensemble classifier based on Support Vector Machines and K Nearest Neighbour Classifiers. A metaheuristic algorithm is further used to optimise the ensemble model.

The construction of the proposed model SCMVE is illustrated in Fig. 1.

3.1 Unsupervised clustering using SOM

The first step of SCMVE is to cluster the entire dataset based on the similarity of attributes between data points. The rationale behind this step is to be able to predict and identify unique characteristics distinct to each cluster.

For this step, we use Self Organising Maps (SOM). SOM can find better sets of clusters on the given dataset (Gholamian et al. 2013).

ML algorithms can be of two types: Supervised and Unsupervised learning. In supervised learning, we predict or categorize the outcome based on the input, and in unsupervised learning, we describe patterns in input data without having the

Table 1 Different classification techniques in credit scoring

Classification method	Description	Advantages	Disadvantages
Singular model classification	Involves conducting supervised classification and training on a singular classification model, such as a neural network or a K-means classifier	Reduced training time. Simple classification models	Low accuracy compared to more complex models
SOM Model for optimization	Involves the usage of a Self Organizing Map for clustering the data points and using a supervised classifier on the optimized data set. SOM may be used in coordination with other optimisation techniques	The clusters found by SOMs are able to generate higher accuracy	Performance decreases with increase in dataset size
Metaheuristic models for optimization	This method involves using metaheuristic optimization models (such as sailfish optimizer or the genetic algorithm) for tasks such as feature selection, weights optimization etc to improve the efficiency of supervised classifiers	Reduced training time. Reduced computational complexity	Performance not consistent: dependent on the task, efficiency metric and other factors
Ensemble Classifiers	This technique involves using multiple base classifiers to produce the output. These classifiers may be optimized using various techniques. The output of these base classifiers is combined using a weighted voting approach	Higher predictive accuracy	High cost of training the models

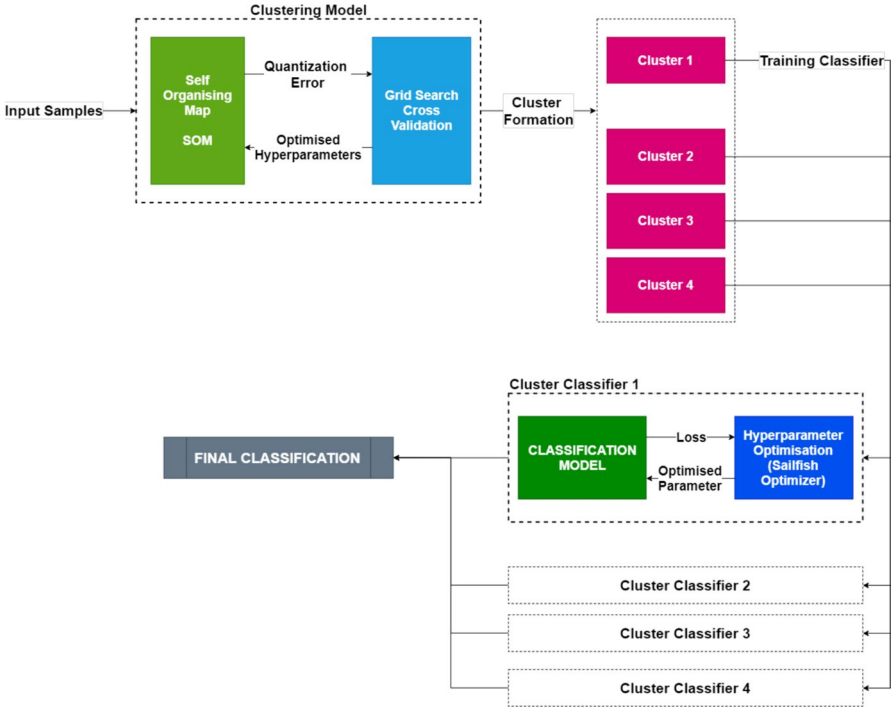

Fig. 1 SOM clustering and metaheuristic voting ensemble classifiers for credit scoring

knowledge of existing class labels. Clustering using Self Organising Maps is an unsupervised learning task.

SOM (Kohonen 1998) is a widely applied neural network-based clustering technique inspired by the arrangement of biological neurons. It is commonly used to represent high dimensional input data into a lower dimension, nonlinear approximation of high dimension data(called the training data), clustering, and feature selection.

As any neural network, a SOM consists of input and output layers. The objective is to transfer all input data to output in such a way that they all connect. Self-organizing maps can overcome the limitations of other statistical techniques through *straightforward implementation, better execution speed, and shorter training periods*. SOM differs from other neural networks in the use of *competitive learning* rather than error correction learning such as backpropagation. These characteristics have resulted in many studies in the past two decades.

Definition 1 (Neuron) Self Organizing maps consist of a grid of nodes. Each node is referred to as a neuron. Each neuron is assigned a weight vector of the dimension of its input vector X_0.

Definition 2 (Best Matching Neuron, ξ) In every iteration, the distance $d_{X \to n}$ between the input vector X_0 and each neuron is calculated.

$$d_{X \to n} = X_0 - n^2 \ \forall n \in X_1, \tag{1}$$

where $X_{1,}$ represents all the layers excluding the input layer. The neuron with the minimum distance is called the best matching neuron ξ and its weight vector is updated along with its neighbors to move it closer to the input vector.

$$\xi = min(d_{X \to n}) \tag{2}$$

Definition 3 (Neighborhood Radius Function, η)—Neighborhood Radius Function is a function used to compute the radius for defining the neighbors of a given neuron. It is a decreasing time function such that the number of neighbors reduces during the training phase. The neighbourhood radius function is represented as:

$$\eta(u, v, S, i) = exp\left(-\frac{d_{u,v}^2}{2\sigma(i)^2}\right), \quad i = 0, 1, 2, ... \tag{3}$$

where u, v represent two vectors of same dimension, S represents the input space, i represents the ith iteration.

Definition 4 (Neighbor Neuron, NN)—The neurons close to the best matching neuron that lie within a certain minimum neighborhood radius NR_{min} are termed as Neighbour Neuron NN which is mathematically represented as

$$NN = \left\{ n \text{ if } \eta(\xi, n, S, i) < \mathrm{NR}_{\min}, \forall n \in S \right. \tag{4}$$

where η is the neighbourhood function, and S represents the entire space/network of the SOM.

Definition 5 (Learning Rate, λ)—Learning rate is a parameter used to update the weights of the neurons.

Definition 6 (Topology-Preservation)—refers to the condition where the mapping preserves the relative distance between nodes. Nodes that are near each other in the input space are mapped to nearby units in the output.

Algorithm 1 Clustering using Self Organising Map

Data : $\boldsymbol{X}$: Training Feature Set,
$\boldsymbol{X_{cluster}}$: Testing Feature Set
SOM.shape: shape of SOM,
$\boldsymbol{\sigma}$: radius of different neighbours,
$\boldsymbol{\lambda}$: learning rate,
num_iter: number of iterations
$\boldsymbol{\eta}$: neighbourhood radius function
Result : **cluster:** List of cluster IDs assigned to each row of feature set

begin
  Initialization
  $i \leftarrow 0$
  $W[num\_iter] \leftarrow \phi$
  $W[0] \leftarrow rand.matrix(SOM.shape)$ /* $0 \leq W[k] \leq 1$ */
  while $i < num\_iter$ **do**
    $selected\_node \leftarrow \phi$
    $min\_dist \leftarrow \infty$
    $min\_node \leftarrow \phi$
    for *randomly selected vector* $v \in X$ **do**
      $d \leftarrow \eta(v, W[i])$ /* Compute distance with weight matrix */
      if $d < min\_dist$ **then**
        $min\_dist \leftarrow \min(d, min\_dist)$
        $min\_node \leftarrow v$
      end
    end
    $W[i+1] \leftarrow W[i] + min\_dist \cdot \lambda \cdot (min\_node - W[i])$ /* Updating weights */
    $i \leftarrow i + 1$
  end
  $y \leftarrow \phi$
  foreach *vector* $v \in X\_cluster$ **do**
    $d_v \leftarrow \eta(v, W[num\_iter])$ /* Store the neighbourhood radius value of a vector */
    $\xi \leftarrow argmin$d_v, cluster_nodes /* Find the winner neuron to assign cluster */
    y.insert(ξ)
  end
end

SOM is commonly set up as a 2D lattice representing a grid-like structure for neurons forming a topological map as described in Fig. 2.

The clustering algorithm is explained below:

1. Each neuron is initiated with a random weight vector, with the same dimensionality as the input vector. Neurons are adjusted during the training period based on competitive learning.

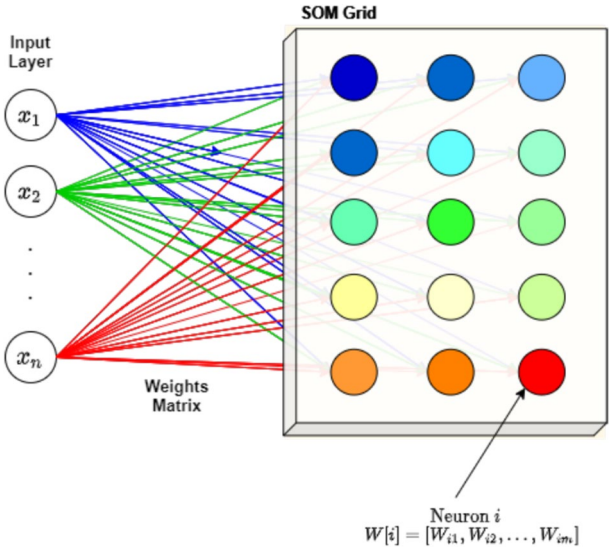

Fig. 2 Self organising map

2. In the SOM grid, every input data is related to every output neuron on the grid. In every iteration i, the neuron with the shortest distance to the input vector is denoted as ξ (or Best Matching Neuron) and the representative vector of ξ is updated so that it moves closer to the input vector.
3. Then, the weights of ξ's neighborhood neurons are updated so that they are symmetrically adapted towards the training input vector. The neuron weights are updated according to the following equation-

$$W_{i+1} = W_i + \eta_{BMU} \cdot \lambda\left(X_i - W_i\right) \tag{5}$$

where X_i is the input vector, W_i is the neuron vector, λ denotes the learning rate, and η_{BMU} represents neighborhood function value for the best matching unit.
4. This training cycle is repeated for each input vector. After finishing the training, the final grid represents the topology-preserving mapping of input data where related neurons are closer to each other.

Note that the update rate for the best matching neuron, the neighbor neuron, and the farther neighbor neuron will differ. The neighborhood radius is a predefined parameter that gradually decreases during the entire training phase and is finally set to 1. A summary of the different parameters involved in the self-organising map is provided in Table 2.

Finally, clusters of neurons are formed, each representing a different class of data that can be appropriately labeled.

Table 2 Parameters involved in the self organising map

Parameter name	Parameter description
η	Neighbourhood radius function
$d_{X \to n}$	Distance computation function
SOM.shape	Shape of the map kernel
λ	Learning rate
num_iter	Number of total training iterations

3.2 Ensemble-based classification model

In this phase of SCMVE, the clusters formed in the first stage are individually used to train and test our ensemble model. Data points in each cluster are split into training and testing data.

The conventional idea is to use a single classifier or a simple combination to make predictions that are often less accurate. One of the techniques to improve the performance of the individual classifier is to combine them using different techniques.

The technique we use is called **ensemble learning**. Various techniques like voting can be used to combine these classifiers used in the ensemble. Ensemble models gather the decisions of several base classifiers trained on the same dataset to get a more effective and accurate decision. The aim is to compensate for the error of a single classifier by other classifiers of the ensemble, increasing the accuracy of the ensemble model as compared to a single classifier model.

For credit scoring, diverse base classifiers could improve the model's accuracy and also provide insightful information about the applicant even if they belong to the same category. The ensemble model used in our paper consists of 12 base classifiers—6 SVM's and 6 KNN's. To ensure maximum utilization and greater diversity of base classifiers, we have used different hyperparameters for each classifier. The classifiers used in the model are explained below -

- **SVM**: an effective classifier model that maps the data into high dimensional spaces and finds an optimal hyperplane to maximize the distance between different classes. A hyperplane is a surface that partitions data points into classes, where points from different classes lie on different sides of the hyperplane. Kernel functions such as linear, polynomial, RBF, and sigmoid are used to map the data into high dimensional spaces.
- **KNN**: one of the most simple yet effective non-parametric classification models. It is an instance-based learning technique and does not require a learning phase. For a data point n to be classified its neighborhood is retrieved i.e k nearest neighbors of n. Selecting an appropriate value of k is necessary as the success of the classifier depends on k.

Votes from each classifier can be combined using several techniques, majority vote ensemble being the most common. In majority voting, each classifier gives a binary vote of 0 and 1. The majority voting ensemble uses two techniques: hard voting and soft voting.

Hard voting makes the final prediction using majority votes from the classifiers, whereas in **soft voting** each classifier corresponds to a probabilistic outcome. The class with the highest probability is predicted as the final output. **In our model, we use the soft voting approach**. The following equations describe the soft voting procedure

Among the n classifiers, each classifier C_i gives the probability P_i such that the predict label $y_i = 0$ for the corresponding input X.

$$P_i = P\big(C_i[X] = 0\big) \tag{6}$$

For each classifier, we assign a weight w_i which determines its relative importance among the others in the ensemble and are assigned based on the performance of an individual classifier (further optimised using the sailfish algorithm as explained later). The final output label y is determined as follows

$$y = \begin{cases} 1 & \frac{\sum_{i=1}^{n} w_i \cdot P_i}{\sum_{i=1}^{n} w_i} \leq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Algorithm 2 Ensemble Model using Soft Voting Classifier

Data : $\boldsymbol{X}$: Input data
$\boldsymbol{N}$: Total classifiers ensembled (in our case $= \mathbf{6}$),
$\boldsymbol{P}[\boldsymbol{N}]$: probability of outcome 0 for each classifier $\boldsymbol{C_i}$
$\boldsymbol{w}[\boldsymbol{N}]$: Array storing weight of each classifier $\boldsymbol{C_i}$
Result : y: Output labels $\in \{0, 1\}$

```
begin
    Initialization
    y ← φ
    P_0 ← 0
    P_1 ← 0
    foreach x ∈ X do
        for n ← 1 to N do
            P_0 = P_0 + P[n] * w_n
            P_1 = P_1 + (1 − P[n]) * w_n
        end
        if P_1 > P_0 then
            y.insert(1)
        end
        if P_0 > P_1 then
            y.insert(0)
        end
    end
end
```

3.3 Sailfish algorithm for weight optimization

In our model SCMVE, we have used a metaheuristic optimization technique, **sailfish optimizer** to generate relative weights for each classifier in the ensemble. The generation of population of weights is done using a uniform distribution using the latest metaheuristic models.

In each epoch of the metaheuristic model training, we find the optimal weight set that can be used for our ensemble model. It is further tested on the train test data and therefore increases the overall efficiency of our model.

- Sailfish optimizer algorithm is a nature-based metaheuristic optimization algorithm recently developed by Shadravan et al. (2019). The algorithm is inspired by the behaviour of sailfish which are known for their coordinated and intelligent hunting strategy (Fig. 3).
- One of the common strategies for group hunting is the attack alternation strategy which is described below:
  - Sailfish usually attack one at a time using their rostrum injuring the sardines severely.

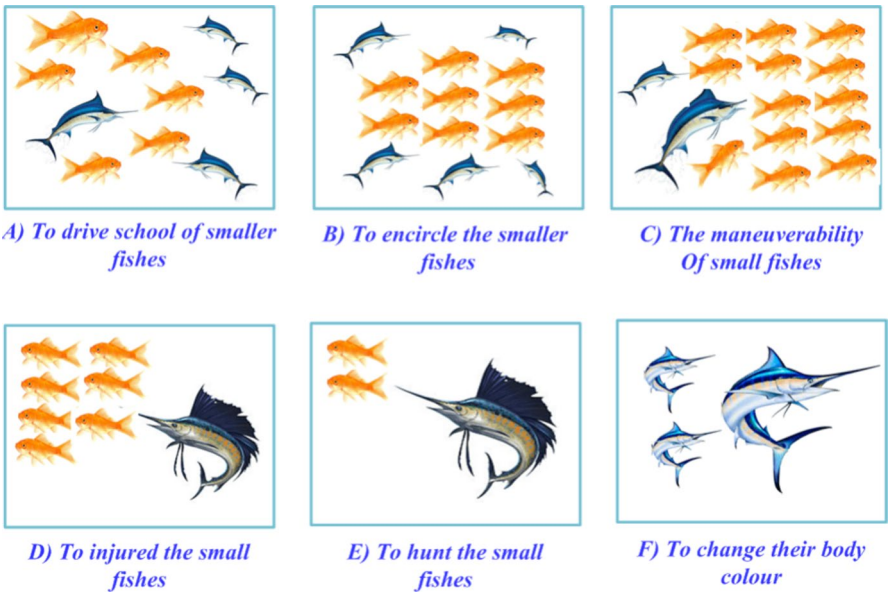

Fig. 3 Sailfish group hunting (Transpire Online sailfish optimizer 2019)

– Being one of the fastest marine beings in the ocean, it is almost impossible for sardines to avoid the attack.
– The injured will eventually be separated from the rest of the prey school and will be captured by the hunters (sailfish).
– Sailfish attacks do not kill the sardine but the frequent attack increases the number of injured sailfish.
– Sardines are another important inspiration for the SFO algorithm. It imitates their ability to change position to escape from an attack.

When tested with other metaheuristic algorithms: Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Genetic Algorithm (GA), Ant Lion Optimizer (ALO), Salp Swarm Algorithm (SSA), and Satin Bowerbird Optimizer (SBO) using a set of unimodal, multi-modal and fixed dimension multimodal benchmark functions, SFO showed competitive results in terms of exploration and exploitation phases (Nassef et al. 2021).

Moreover, SFO shows high-speed convergence on multimodal functions in reaching the global optimum while avoiding the local optimum.

Definition 7 (Search Space, S) It is the set of all possible solutions that the input can take. It contains a set of points (each representing a possible solution) that give the optimal solution. Optimization aims to search that point(solution).

Definition 8 (Fitness Function, $\psi(x)$) It is an objective function that defines the optimality of a solution to a given target problem. It defines how close a given solution is to achieve the desired output. Any mathematical operation that is able to assign computable scores to the states of the matrices can be used for the approach.

The functions that have been experimented with in this paper are mentioned in Table 3. For this paper, the mean squared error function is used to compute the fitness as per the Algorithm 3.

Table 3 Description of fitness functions

Function	Abbreviation	Formula
Mean squared error	MSE	$\text{MSE}(x,y) = \frac{1}{n}\sum_{i=0}^{n-1}(x_i - y_i)^2$
Mean absolute error	MAE	$\text{MAE}(x,y) = \frac{1}{n}\sum_{i=0}^{n-1}\lvert x_i - y_i\rvert$
Mean square logarithmic error	MSLE	$\text{MSLE}(x,y) = \frac{1}{n}\sum_{i=0}^{n-1}(\log_e(1+x_i) - \log_e(1+y_i))^2$
Mean tweedie deviance (order = 1)	MTD-1	$\text{MTD}_1(x,y) = \frac{1}{n}\sum_{i=0}^{n-1} 2(x_i \log(x_i/y_i) + y_i - x_i)$
Mean tweedie deviance (order = 2)	MTD-2	$\text{MTD}_2(x,y) = \frac{1}{n}\sum_{i=0}^{n-1} 2(\log(y_i/x_i) + x_i/y_i - 1)$
Mean tweedie deviance (order = 3)	MTD-3	$\text{MTD}_3(x,y) = \frac{1}{n}\sum_{i=0}^{n-1} 2\left(\frac{\max(x_i,0)^{-1}}{2} + \frac{xy_i^{-2}}{2} - y_i^{-1}\right)$
Mean tweedie deviance (order = 4)	MTD-4	$\text{MTD}_4(x,y) = \frac{1}{n}\sum_{i=0}^{n-1} 2\left(\frac{\max(x_i,0)^{-2}}{6} + \frac{xy_i^{-3}}{3} - \frac{y_i^{-2}}{2}\right)$

Algorithm 3 Fitness Function Computation using Mean Squared Error

Data : $\boldsymbol{N}$: Number of classifiers
$\boldsymbol{W}[\boldsymbol{N}]$: Weight array containing the weights for the classifiers
$\boldsymbol{X_{train}}$: Training data
$\boldsymbol{y_{train}}$: Training labels
$\boldsymbol{X_{test}}$: Testing set
$\boldsymbol{y_{test}}$: Testing Labels
Result : $\boldsymbol{\psi(x)}$: Fitness value for the given weight array

begin
  Initialization
  $P[N] \leftarrow \phi$ /* Array to store the predictions of the classifiers */
  Initialise classifiers $C_{1 \rightarrow N}$
  $M \leftarrow \text{Ensemble}(C_i \quad \forall i \in \{1 \ldots N\})$ /* Combines the individual classifiers into an ensemble */
  $M.\text{train}(X_{train})$
  foreach $x \in X_{test}$ **do**
    $p \leftarrow M.\text{predict}(x)$
    $P.\text{insert}(p)$
  end
  $\psi \leftarrow \frac{1}{\text{length}(y_{test})} \cdot y_{test} - P^2$ /* Sum of squared error */
end

Definition 9 (Prey Density, PD) The value indicating the number of prey(sardines) present in each iteration and it decreases in every iteration as the prey decreases during hunting. The value PD can be calculated as follows

$$PD = 1 - \left(\frac{N_{SF}}{N_{SF} + N_S} \right) \tag{8}$$

where N_{SF} and N_S are the numbers of sailfish and sardines in each cycle, respectively.

Definition 10 (Attack Power, AP) Represents the sailfish's attack power at each iteration. It helps us to calculate the number of sardines that update their position. The attack power is calculated as

$$AP = A \times (1 - (2 \times i \times \varepsilon)) \tag{9}$$

where A represents the coefficient of the sailfish's decrementing attack power, ϵ is the learning rate and i denotes the ith iteration.

Definition 11 (Coefficient of Mutation of Position, θ_P) At each step of the algorithm, the positions of the sailfish and sardines change. The extent to which such a positional change occurs is given by the coefficient of mutation of position, θ_P which is analogous to a time-varying learning rate.

The expression for the coefficient is

$$\theta_P = PD \cdot (2 \cdot \text{rand}(0, 1) - 1) \tag{10}$$

The sailfish are assumed as candidate solutions and their position in the search space denotes the problem variables.

For n sailfish and m sardines in the optimization problem of d dimensions, matrix $P^i_{sf}[n][d]$ is used to show the position of every sailfish at the ith iteration.

Sardines are assumed to be swimming in the same search space. Matrix $P^i_{sar}[m][d]$ shows the position of every sardine at the ith iteration.

To compute the fitness value of each sailfish x and each sardine y the objective fitness function is applied to each row at a given iteration i as follows

$$\begin{aligned} &\text{fitness value of sailfish x} = \psi(P^i_{sf}[x][,]) \\ &\text{fitness value of sardine y} = \psi(P^i_{sar}[y][,]) \end{aligned} \tag{11}$$

The following matrices give the fitness value for all positions of sailfish and sardines

$$\Psi_{sf}[n][1] = \begin{bmatrix} \psi(P^i_{sf}[0][0,d]) \\ \psi(P^i_{sf}[1][0,d]) \\ \vdots \\ \psi(P^i_{sf}[n][0,d]) \end{bmatrix} = \begin{bmatrix} \Psi_{SF}[0] \\ \Psi_{SF}[1] \\ \vdots \\ \Psi_{SF}[n] \end{bmatrix} \tag{12}$$

where n is the number of sailfish and d shows the number of variable and $P^i_{sf}[x][,]$ is the position of sailfish x

$$\Psi_{sar}[m][1] = \begin{bmatrix} \psi(P^i_{sar}[0][0,d]) \\ \psi(P^i_{sar}[1][0,d]) \\ \vdots \\ \psi(P^i_{sar}[m][0,d]) \end{bmatrix} = \begin{bmatrix} \Psi_{sar}[0] \\ \Psi_{sar}[1] \\ \vdots \\ \Psi_{sar}[m] \end{bmatrix} \tag{13}$$

where m is the number of sardines and d shows the number of variables and $P^i_{sar}[y][,]$ is the position of sailfish y.

Algorithm 4 Sailfish Optimization

Data : $\boldsymbol{A}$: Coefficient of sailfish attack power,
$\boldsymbol{n}$: Number of sailfish,
$\boldsymbol{m}$: Number of sardines,
$\boldsymbol{d}$: Dimensionality of data,
$\boldsymbol{\epsilon}$: Learning rate,
$\boldsymbol{\psi(x)}$: Objective function
Result : $\boldsymbol{SF_{best}}$: Position of best sailfish

begin
  Initialization
  $P_{sf}^{1000}[n][d] \leftarrow \text{rand}(0,1)$
  $P_{sar}^{1000}[m][d] \leftarrow \text{rand}(0,1)$
  $F_{sf}[n][1] \leftarrow \Psi(P_{sf})$
  $F_{sar}[m][1] \leftarrow \Psi(P_{sar})$
  $SF_{best} \leftarrow \text{rand}(0,n)$
  $SAR_{best} \leftarrow \text{rand}(0,m)$
  $D_{prey} \leftarrow 1 - \frac{n}{n+m}$
  $i \leftarrow 0$
  while *true* **do**
    foreach *sailfish* $x \leftarrow 0$ *to* n **do**
      $\theta_P \leftarrow D_{prey}(2 \cdot \text{rand}(0,1) - 1)$
      $P_{sf}^{i}[x] \leftarrow SF_{best} - \theta_P \left(\text{rand}(0,1) \cdot \frac{SF_{best}+SAR_{best}}{2} - P_{sf}^{i}[x]\right)$
    end
    $AP \leftarrow A \cdot (1 - 2 \cdot i \cdot \epsilon)$
    if $AP < 0.5$ **then**
      $\alpha \leftarrow m \cdot AP$
      $\beta \leftarrow d_i \cdot AP$
      $S_{selected} \leftarrow$ list with α number of 1 and $m - \alpha$ number of 0
      $d_{selected} \leftarrow$ list with β number of 1 and $d - \beta$ number of 0
      $S_{sar} \leftarrow P_{sar}^{i}[S_{selected}][d_{selected}]$
      $S_{sar} \leftarrow\sim \mathcal{N}(0,1) \cdot (SF_{best} - S_{sar} + AP)$
    end
    else
      $P_{sar} \leftarrow\sim \mathcal{N}(0,1) \cdot (SF_{best} - P_{sar} + AP)$
    end
    $F_{sar}[m][1] \leftarrow f(P_{sar})$
    if $\Psi(P_{sar}^{i}) < \Psi(P_{sf}^{i})$ **then**
      $P_{sf}^{i} \leftarrow P_{sar}^{i}$
      P_{sar}^{i}.remove(SAR_{best})
    end
    $i \leftarrow i + 1$
  end
end

Algorithm 5 SCMVE Credit Scoring

Data : $\boldsymbol{X_{train}}$: Training feature set,
$\boldsymbol{y_{train}}$: Training feature labels
$\boldsymbol{X_{test}}$: Testing feature set
$\boldsymbol{SOM_c}$: SOM clustering model
$\boldsymbol{E_c}$: Ensemble classification model
$\boldsymbol{S(A, n, m, \epsilon, \phi)}$: Sailfish weight optimizer
Result : $\boldsymbol{m}$: Trained SOM model,
$\boldsymbol{y_{test}}$: Predicted labels for $\boldsymbol{X_{test}}$

begin
  SOM_c.train(X_{train}) /* clusters will be treated as the new class labels */
  $C \leftarrow SOM_c$.cluster_labels
  $CM_{\text{len(C)}} \leftarrow \phi$ /* Pipeline for classification */
  $N \leftarrow len(E_c)$
  foreach *cluster* $c \in C$ **do**
    $w[N] \leftarrow$ rand($list(N)$) /* Initialise a randomized weight list for ensemble classifier */
    $w[N] \leftarrow S($
      $A = 4$
      $n = 100$
      $m = 25$
      $\epsilon = 0.001$
      $\psi(N = 6, w[N], X_{train},$
      $y_{train}, X_{test}, y_{test})$
) /* Optimise the weights of the ensemble using Sailfish Optimiser S */
    $CM_c \leftarrow E_c$.train(
      X_{train}[cluster $= c$],
      weight $= w[N]$
) /* Train on the optimised weights */
  end
  /* Training ends */
  $y_{test} \leftarrow \phi$
  foreach $x \in X_{test}$ **do**
    $c_{pred} \leftarrow SOM_c$.predict($x$)
    prediction $\leftarrow CM_{c_{pred}}$.predict(x)
    y_{test}.insert(prediction)
  end
end

The sailfish with the best fitness values are considered elite and are best suited for attacking the prey. These are the sailfish that do not change their position in upcoming iterations until the optimal solution is not lost. Moreover, sardines with the best fitness function are considered the best target by sailfish and are the most exposed to injury.

The sailfish optimizer, given the above-mentioned definitions, works as follows

1. An arbitrary sailfish is assigned to be an elite sailfish whose position is denoted by SF_{best} and an arbitrary sardine the injured sardine whose position is SAR_{best}.
2. The prey density value is calculated as per Eq. 8.
3. The coefficient of mutation of position λ_p is calculated as per Eq. 10.
4. For each sailfish, its position is updated by a certain magnitude as represented in the following equation

$$P^i_{sf}[x] = SF_{best} - \theta_P\left(\text{rand}(0,1) \cdot \frac{SF_{best} + SAR_{best}}{2} - P^i_{sf}[x]\right)$$

5. As the hunting continues through each iteration, the power of the sailfish's attack also reduces as given in Eq. 9. Further, sardines will have less energy and will not be able to properly detect the sailfish position, hence the propensity of getting injured increases. Hence, the sardine positions are also updated as follows

$$P_{sar} = \sim \mathcal{N}(0,1) \cdot (SF_{best} - P_{sar} + AP) \tag{14}$$

where $\sim \mathcal{N}(0,1)$ denotes a random generated between 0 and 1.

6. Next, to determine the number of sardines updating their positions α, we make use of the following expression

$$\alpha = N_s \times AP \tag{15}$$

where N_s denotes the number of sardines.

7. In addition to this, the number of variables updated β, for the corresponding sardine updation is given as

$$\beta = d_i \times AP \tag{16}$$

where d_i is the number of variables in i_{th} iteration

8. The updation occurs as per the following conditions

   (a) If $AP < 0.5$, then positions of only α sardines are updated
   (b) Else every sardine updates its position.

9. Finally, the attack is initiated, and the injured sardines are captured. It is assumed to be the case that the sardines are attacked if their fitness values are better than the sailfish.
10. The space of the captured sardine is substituted by sailfish to increase the chances of hunting. The conditional substitution is represented by the following expression

$$P^i_{sf} = P^i_{sar} \quad \text{if} \quad \Psi(P^i_{sar}) < \Psi(P^i_{sf}) \tag{17}$$

where P^i_{sar} is the position of sardine at its ith iteration and P^i_{sf} gives the position of sailfish at ith iteration.

11. This cycle of updation of positions of sardines and sailfish continues until the end criterion is fulfilled.

The different parameters involved while performing ensemble weight optimization using the Sailfish Algorithm are give in Table 4. The weights generated from the optimization are used with results from the 6 individual classifiers to generate the final predictions.

The last step is to predict the final accuracy of SCMVE, which is calculated using the weighted average of the accuracy of individual classifiers. We use a weighted voting technique and a sailfish optimizer for weight generation for the following reasons:

- Relative weighing the base classifiers on the basis of their performance helps us to value the classifiers that perform better on a certain cluster and undervalue those that perform worse. This helps us to increase our overall performance. In comparison, if all base classifiers were weighed equally, it would lead to a low performing base classifier reducing the overall accuracy of the model.
- Generating the weights using a metaheuristic optimizer converts this into an optimization problem. This not only reduces the training time since we can traverse the search space faster given the nature of our optimizer and its exploration capabilities, it also helps us reach a global maxima on the model efficiency, which otherwise can lead to models getting stuck on local maxima due to inefficient search space exploration strategies.

The final model metrics can be calculated as:

$$NN = \left\{ n \text{ if } \eta(\xi, n, S, i) < \mathrm{NR}_{\min}, \forall n \in S \right. \tag{18}$$

where C_i denotes the ith cluster, X denotes the input dataset and P denotes the performance metrics, for instance, accuracy.

Table 4 Parameters involved in the sailfish optimizer

Parameter name	Parameter description
P	Size of the sailfish population. Denotes the number of solution arrays formed to search the vector space
ϵ	Learning rate of algorithm
N	Number of base classifiers in the ensemble
Lower Bound (lb)	Lower bound of weight search space
Upper Bound (ub)	Upper bound of weight search space
Fitness Function (Φ)	Transformation/function to evaluate the fitness value of given populations

4 Experimental setup

4.1 Description of credit datasets

Security and confidentiality concerns restrict us from accessing private real-world credit scorinf datasets from banks. However, the UCI Machine Learning Repository has various real-world public datasets for credit scoring that are commonly utilized in many research papers to evaluate the performance of the proposed algorithms. We use three public credit datasets from the UCI Machine Learning Repository (Dua and Graff 2017) to evaluate the performance of our proposed approach: Australian Dataset and German Dataset and Taiwan Dataset. The three datasets utilized to evaluate the performance of our proposed model are described in detail in Table 5. The Australian dataset (AUS) has 690 instances, 307 of which are in the good category and 383 in the bad category. There are 14 attributes in the dataset. The German dataset (GER) contains 1000 instances, 700 of which are in the good class and 300 in the bad class. There are 24 attributes in the dataset, including the purpose, credit amount, and personal information. The Taiwan dataset (TAI) has 30,000 instances, 23,364 of which are in the good category and 6636 in the bad category. There are 24 attributes in the dataset. Tables 6, 7 and 8 show a more detailed summary of the attributes.

Table 5 Description of Australian and German and Taiwan credit datasets

Dataset	Australian	German	Taiwan
Number of instances	690	1000	30,000
Number of features	14	20	23
Categorical features	8	13	4
Numerical features	6	7	19
Number of good/bad instances	307/383	700/300	23,364/6636

Table 6 Attribute description of Australian credit dataset

Attribute	Type of attribute	Range
A1	Categorical	0–1
A2	Numerical	13.75−80.25
A3	Numerical	0–28
A4	Categorical	1–3
A5	Categorical	1–14
A6	Categorical	1–9
A7	Numerical	0−28.5
A8	Categorical	0–1
A9	Categorical	0–1
A10	Numerical	0–67
A11	Categorical	0–1
A12	Categorical	1–3
A13	Numerical	0–2000
A14 Class variable	Numerical	1–100,001
A15	Categorical	0–1

Table 7 Attribute description of German credit dataset

Attribute	Type of attribute	Range
A1	Categorical	1–4
A2	Numerical	4–72
A3	Categorical	0–4
A4	Categorical	0–10
A5	Numerical	250–18,424
A6	Categorical	1–5
A7	Categorical	1–5
A8	Numerical	1–4
A9	Categorical	1–5
A10	Categorical	1–3
A11	Numerical	1–4
A12	Categorical	1–4
A13	Numerical	19–75
A14	Categorical	1–3
A15	Categorical	1–3
A16	Numerical	1-4
A17	Categorical	1–4
A18	Numerical	1–2
A19	Categorical	1–2
A20 Class variable	Categorical	1–2
A21	Categorical	1–2

Table 8 Attribute description of Taiwan credit dataset

Attribute	Type of attribute	Range
A1	Numerical	10,000-1,000,000
A2	Categorical	0–1
A3	Categorical	0–2
A4	Categorical	0–1
A5	Numerical	21–79
A6	Numerical	(–2)–(8)
A7	Numerical	(–2)–(8)
A8	Numerical	(–2)–(8)
A9	Numerical	(–2)–(8)
A10	Numerical	(–2)–(8)
A11	Numerical	(–165,580)–(964,511)
A12	Numerical	(–69,777)–983,931
A13	Numerical	(–157,279)–1,664,089
A14	Numerical	(–170,000)–891,586
A15	Numerical	(–81,334)–927,171
A16	Numerical	(–339,603)–961,669
A17	Numerical	0–873,552
A18	Numerical	0–1,689,259

Table 8 (continued)

Attribute	Type of attribute	Range
A19	Numerical	0–896,040
A20	Numerical	0–621,000
A21	Numerical	0–426,529
A22	Numerical	0–528,666
A23	Categorical	0–1

4.2 Data preprocessing and dimensionality reduction

Data preprocessing is an important step before constructing the proposed model since it improves data interpretability and eliminates the sensitivity of difference classifiers to scale differences in attribute ranges. In the first phase, we select the features using the methods described in Sect. 3. We encoded the categorical variables using the dummy variable approach after feature selection. In the dummy variable approach, a variable with n potential values is transformed into n features with a binary value of either 0 or 1. Numerical features, on the other hand, are normalized by following Eq. 1

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)} \tag{19}$$

where x represents the original feature value before normalization, x_{norm} represents the new feature value after normalization, and $min(x)$ and $max(x)$ represent the original feature's minimum and maximum feasible values, respectively. This reduces the sensitivity of different classifiers by scaling each feature's value to a range between 0 and 1.

4.3 Model evaluation metrics

We detail the various performance evaluation measures we used to evaluate our proposed model and compare its performance with that of other well-known classifiers in this section.

Five evaluation metrics are used to assess the overall classification performance of our proposed model: accuracy, precision, recall, F1 score and AUC. The most commonly used evaluation metric is accuracy, which is defined as the ratio of correctly identified instances to the total number of instances in the dataset. However, in the event of a dataset with a class imbalance, accuracy does not reflect the model's true performance

$$Accuracy = \frac{TP + FN}{TP + FN + FP + TN} \tag{20}$$

Another widely used metric in credit scoring studies is the AUC, which is based on the receiver operating characteristic (ROC) curve. The ROC curve is a visualisation of the true positive rate (TPR) and false positive rate (FPR) in binary classification

tasks (FPR). AUC, which is defined as the area under the ROC curve, is used to compare different classifiers. TPR and FPR are defined as follows:

$$TPR = \frac{TP}{TP + FN} \tag{21}$$

$$FPR = \frac{FP}{FP + FN} \tag{22}$$

The following are the definitions for the last three metrics: precision, recall, and F1 score:

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

$$F1\ score = 2 * \frac{precision * recall}{precision + recall} \tag{25}$$

5 Experimental results

We exhibit the obtained data to strengthen the claim that the ensemble model formulated in this study outperforms the other benchmark models in the important parameters mentioned in this paper. Experiments mentioned in this study were conducted with Python 3.7 on Google Colab. Various classifier Table 9 results are compared with our proposed technique, represented in Figs. 4, 5, 6, 7 and 8 in terms of accuracy, precision, recall, F1 score and AUC. In Sect. 5.1, we present a comparative analysis of the results of numerous individual classifiers. Section 5.2 includes the comparison of our method with other classical ensemble methods. In Sect. 5.3, the result of SCMVE is highlighted, and we compare it with various individual and ensemble classifiers.

5.1 Classification results of individual classifiers

In this section, we evaluate the performance of the individual classifiers viz. Logistic Regression, Naive Bayes, K Nearest Neighbors, Gradient Boosting, Decision Trees, Support Vector Machine and Multi Layer Perceptron(MLP) in their ability to evaluate good credit from bad credit based on the features input into the models. Results of the individual classifiers on the Australian dataset can be seen from Table 10. Upon examination, it shows that the performance of all individual classifiers on the performance metrics—Accuracy, Recall and F1 score varies highly. Naive Bayes and K Nearest Neighbours show a relative

Table 9 Values of parameters for different classifiers

Classifier	Parameter	Description	Value
Neural network	Hidden_layer_sizes	Count of neurons in the ith hidden layer	100
	Activation	Activation function for the hidden layer	Tanh
2–4	Alpha	Regularization term parameter	0.001
	Learning_rate	Learning rate schedule for weight updates	Adaptive
	Solver	Solver for weight optimization	SGD
K nearest neighbor	n_neighbors	Number of neighbors to use	5
Support vector machine	C	Regularization parameter	1.0
	Kernel	Kernel type	Poly
	Probability	Enable probability estimates	True
	Gamma	Kernel coefficient	0.1658 (Australian dataset), 0.1221 (German dataset), 0.02631 (Taiwan dataset)
Random forest	n_estimators	Number of trees in the forest	200

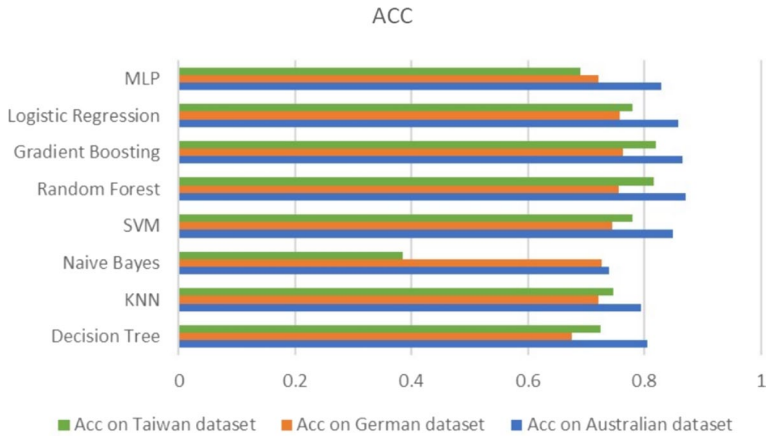

Fig. 4 Comparison of accuracy

decline in performance. Another observation is that the performance of Random Forest Algorithm is significantly better than the rest of the individual classifiers in terms of metrics—accuracy, precision and F1 score, Support Vector Machine in recall. Results for Taiwan dataset are shown in Table 11. For the

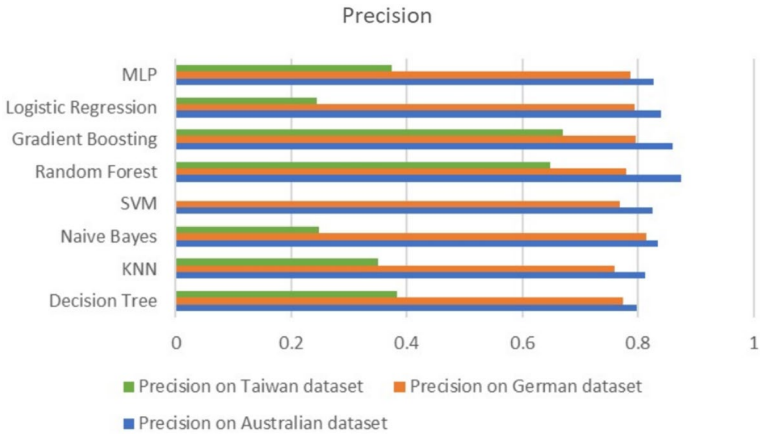

Fig. 5 Comparison of precision

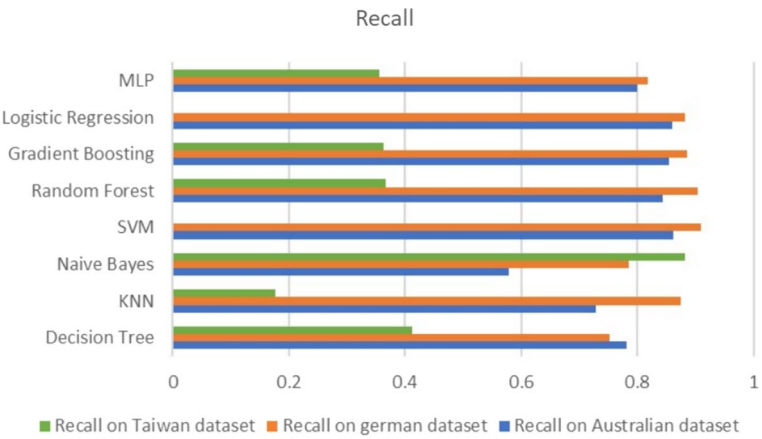

Fig. 6 Comparison of recall

German Credit dataset, the performance of all individual classifiers as tested in our experiments is displayed in Table 12. The following inferences can be made from observations of the models' performance: Gradient Boosting and Random Forest outperform the rest of the classifiers on accuracy and F1 score metrics. Logistic Regression and Random Forest perform well in terms of precision, recall, and AUC as well. In tandem with the Australian dataset, the Random Forest model outperforms others in the precision metric score, while the SVM model outperforms on the recall metric on the German Dataset.

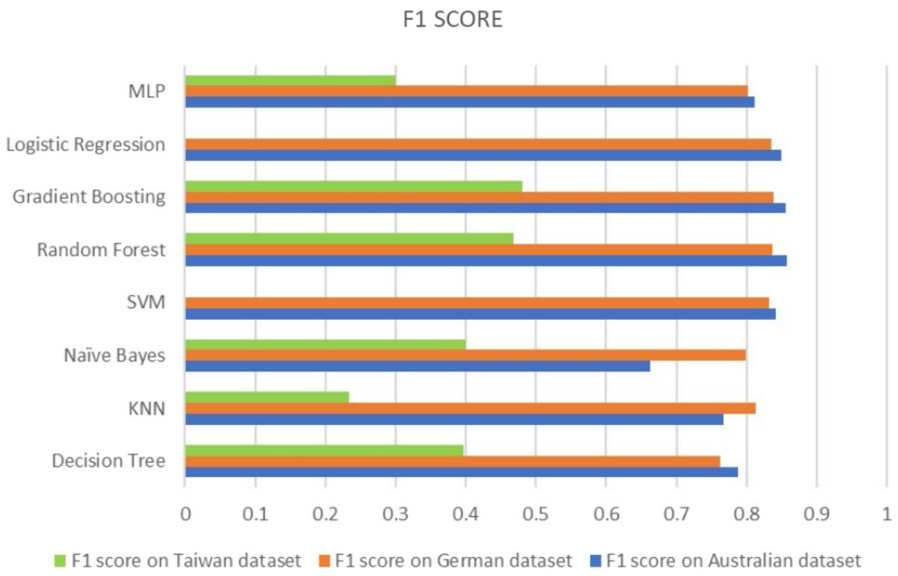

Fig. 7 Comparison of F1 Scores

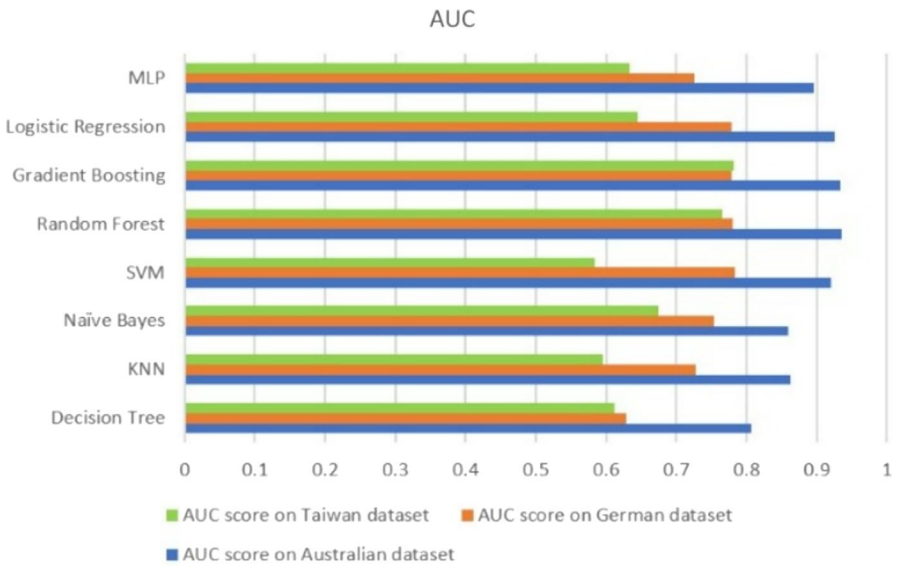

Fig. 8 Comparison of AUC

For Taiwan dataset, the Gradient Boosting model outperforms others in the precision metric score, while the Naive Bayes model outperforms on the recall metric of both Autralian and German dataset.

Table 10 Comparison of results of different models over Australian Credit Dataset

Algorithm	Accuracy	Precision	Recall	F1 score	AUC
Decision tree	0.805	0.798	0.781	0.788	0.807
KNN	0.794	0.813	0.729	0.767	0.862
Naive Bayes	0.739	0.834	0.579	0.663	0.860
SVM	0.849	0.825	0.861	0.842	0.921
Random forest	0.870	0.874	0.844	0.858	0.935
Gradient boosting	0.866	0.859	0.854	0.856	0.933
Logistic regression	0.858	0.840	0.860	0.850	0.926
MLP	0.828	0.827	0.799	0.812	0.896

Table 11 Comparison of results of different models over Taiwan credit dataset

Algorithm	Accuracy	Precision	Recall	F1 score	AUC
Decision tree	0.724646	0.383566	0.412779	0.397538	0.612760
KNN	0.746465	0.349367	0.175882	0.233901	0.596862
Naive Bayes	0.384040	0.247821	0.882347	0.3868822	0.674081
SVM	0.779838	0.000000	0.000000	0.000000	0.584741
Random forest	0.816545	0.647927	0.365800	0.467442	0.765633
Gradient boosting	0.820323	0.669441	0.363646	0.471151	0.781185
Logistic regression	0.779778	0.245000	0.000228	0.000456	0.645162
MLP	0.689152	0.374586	0.356418	0.299416	0.634567

Table 12 Comparison of results of different models over German credit dataset

Algorithm	Accuracy	Precision	Recall	F1 score	AUC
Decision tree	0.675	0.774	0.752	0.762	0.628
KNN	0.720	0.760	0.874	0.813	0.718
Naive Bayes	0.726	0.814	0.785	0.799	0.753
SVM	0.745	0.768	0.909	0.832	0.784
Random forest	0.755	0.780	0.904	0.837	0.780
Gradient boosting	0.762	0.795	0.886	0.838	0.779
Logistic regression	0.757	0.793	0.882	0.835	0.778
MLP	0.720	0.787	0.8178	0.802	0.725

5.2 Classification results of ensemble classifiers

Individual classifiers yield good results for all the Australian and German along with Taiwan dataset, but most researchers have used an ensemble of individual classifiers. Using a mixture of multiple base learners can assist reducing variance and bias, and hence increase the accuracy of predictions. In this research, we examine the performance of various proposed ensemble approaches, such as Tree-Based

Dynamic Ensemble (Xia et al. 2020), B-Stacking approach (Xia et al. 2018), Semi-Supervised Selective Ensemble (Xiao et al. 2020) and Enhanced Outlier adaptation (Zhang et al. 2021) to show that ensemble classifiers outperform solo classifiers on credit scoring datasets. Tables 10 and 12 demonstrate the results of some popular ensemble approaches on all Australian and German as well as Taiwan datasets. The data shows that all other ensemble classifiers outperform most individual classifiers on most parameters across both datasets. The proposed model approach was the best performer overall.

5.3 Performance of the proposed SCMVE approach

The results in Tables 10 and 12 show that the classification method described in this research performs significantly better than other singular or hybrid ensemble classifier on all the Australian and German as well as the Taiwan dataset. A soft voting metaheuristic approach is proposed by our model. It entails examining the performance of different models by training them on clusters and producing significantly enhanced results.

While the proposed model outshines all other classifiers across the evaluation metrics by a large margin for the Australian dataset (Fig. 9), the signifanct achievement of this paper and the proposed model is the subsequent improvement of performance in the German dataset (Fig. 10) and Taiwan dataset, where our model demonstrates an overall upgradation in results achieved through our cluster-based metaheuristic approach.

Further, on experimentation with the selection of the various fitness functions as referenced in Table 3, we observe a variation in the performance in accordance with the usage of each fitness function as shown in Figs. 11 and 12. This approach of finding the most suitable fitness function has been highlighted in Huang et al.

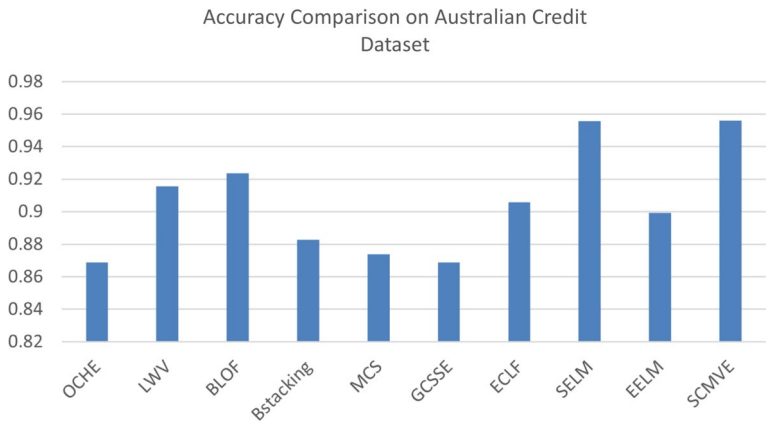

Fig. 9 Comparison of accuracy on Australian dataset

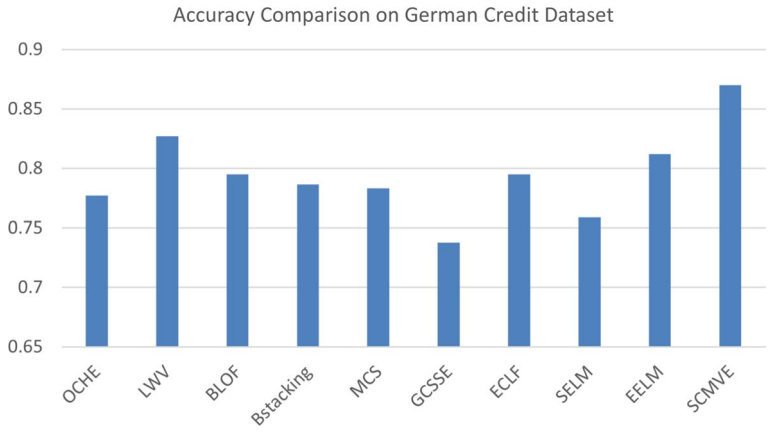

Fig. 10 Comparison of accuracy on German dataset

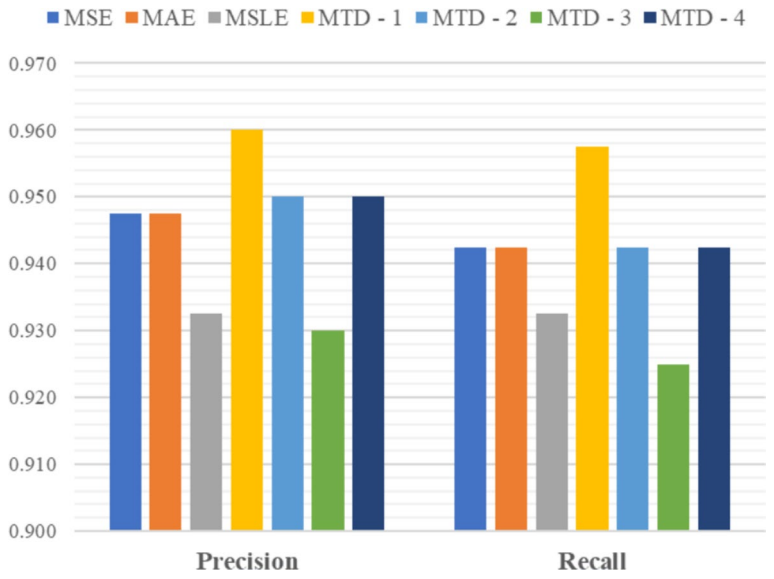

Fig. 11 Variation of precision and recall with the fitness function used on the Australian dataset

(2019) wherein the parameter optimization methods for metaheuristic algorithms have been studied. We observe that although the fitness function MTD-1 performs better in most aspects, it fails to provide a consistent AUC score. Hence, to ensure simplicity of the model and a consistent performance, the fitness function MAE was chosen.

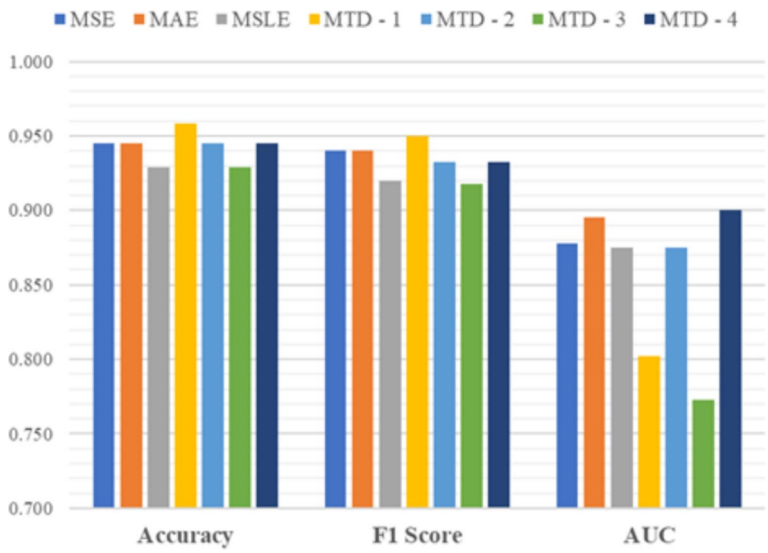

Fig. 12 Variation of accuracy, F1 Score, and AUC Score with the fitness function used on the Australian dataset

5.4 Comparison of SCMVE with other approaches of credit scoring

For credit scoring, a variety of single classifiers and ensemble-based approaches have been developed. On the German and Australian datasets, Table 13 compares a few credit rating techniques. Table 14 on the other hand compares few credit rating techniques over Taiwan dataset. SCMVE clearly outperforms other approaches, such as tree-based dynamic heterogeneous ensemble method, multi-stage ensemble model with enhanced outlier adaptation, credit scoring model based on ensemble feature selection and multilayer ensemble classification and Multi-Stage Self-Adaptive Classifier Ensemble Model.

Further, models like LWV (Tripathi et al. 2019) and BLOF (Zhang et al. 2021) which come close to our model performance on the Australian dataset, fail to show similar state-of-the-art performance in the German Credit Data Set.

Our usage of the recent sailfish optimizer helps us in reducing computational time. This works better in comparison to the works of Xia et al. (2020) where the tree-based architecture can lead to increased computational complexity requiring GPU usage, demands that were not required in our model.

Most existing works, such as that of Tripathi et al. (2019) embed feature selection into their model to increase their model efficiency. This can lead to some feature of data points not being used in the final output, even though they might be representative of the sample space. We overcome this issue through the usage of a Self Organising Map. Our model, instead of using feature selection, sections the data into distinct clusters with their own unique properties and feature representations. We then use our classification model on each cluster so that we utilize all the features

Table 13 Comparison of the performance achieved by various ensemble approaches of credit scoring over Australian and German dataset

Authors	Model description	Accuracy: Australian dataset	Accuracy: German dataset
Xia et al. (2018)	The model proposed differs from the existing ensemble credit models in three aspects-pool generation, selection of base learners, and trainable fuser	0.8828	0.7866
Tripathi et al. (2019)	This model conducts preprocessing and assigns weights to classifiers. Then feature selection using an ensemble model is applied to a classifier framework	0.9155	0.8268
Guo et al. (2019)	Introduces a novel multi-stage self-adaptive classifier ensemble model based on the statistical techniques and the machine learning techniques to improve the prediction performance	0.8740	0.7830
Xia et al. (2020)	This work develops a novel tree-based overfitting-cautious heterogeneous ensemble model (i.e., OCHE) for credit scoring.	0.8689	0.7772
Xiao et al. (2020)	Proposes a cost-sensitive semi-supervised selective ensemble model based on group method of data handling	0.8689	0.7376
Kuppili et al. (2020)	A novel spike-generating function is proposed in Leaky Nonlinear Integrate and Fire Model (LNIF). Its interspike period is computed and utilized in the extreme learning machine (ELM) for classification	0.9558	0.7589
Tripathi et al. (2020)	A novel parametrized algebraic activation function is proposed for extreme learning machine (ELM)	0.8992	0.8118

Table 13 (continued)

Authors	Model description	Accuracy: Australian dataset	Accuracy: German dataset
Liu et al. (2022)	This work develops a Multi-grained and multi-layered gradient boosting decision tree for credit scoring.	0.8826	0.7653
Zhang et al. (2021)	A local outlier factor algorithm is enhanced with bagging strategy to identify outliers and boost them back into the training set to construct an outlier-adapted training set	0.9236	0.7950
Zhang et al. (2021)	A voting-based outlier detection method is proposed to enhance the outlier detection algorithms with the weighted voting mechanism and boost the outlier scores into the training set to form an outlier-adapted training set	0.9058	0.7950
Runchi et al. (2023)	The Logistic-BWE model combines logistic regression with heterogeneous balancing and dynamic weighting. Sub-models are trained on datasets with different imbalance ratios, improving recognition of specific classes. Ensemble weights are dynamically adjusted based on each sub-model's prediction accuracy	0.865	0.757
Our proposed approach (SCMVE)	**SOM-based clustering supplementing a weighted voting ensemble classifier that uses a Sailfish Optimizer for weights generation**	**0.9559**	**0.8700**

Table 14 Comparison of the performance achieved by various ensemble approaches of credit scoring over Taiwan dataset

Authors	Approach	Accuracy: Taiwan dataset
Liu et al. (2022)	Multi-grained and multi-layered gradient boosting decision tree for credit scoring	69.22%
Zhang et al. (2021)	Local outlier factor algorithm with bagging strategy to construct an outlier-adapted training set	82.93%
Zhang et al. (2021)	Voting-based outlier detection method with the weighted voting mechanism to form an outlier-adapted training set	82.26%
Zhou et al. (2023)	This credit risk model for small firms uses a two-stage expert system with SMOTE for imbalanced data and random forest for feature extraction	75.7%
Our proposed approach (SCMVE)	SOM-based clustering supplementing a weighted voting ensemble classifier that uses a Sailfish Optimizer for weights generation.	82.80%

Bold values indicate better results than previous models

of each cluster to generate an output, thus not ignoring any feature that might be distinctly representative in a particular cluster. Credit scoring has outlier issues and most works have to engineer outlier detection algorithms to overcome the situation. Examples of this are present in the works of Zhang et al. (2021, 2021) Our usage of an ensemble classifier, combined with the clusters formed through the SOM helps us overcome outliers in the sample space since the result of multiple base classifiers perform better on outlier detection as opposed to singular classification models.

6 Conclusion and future work

Credit scoring is an important parameter in helping firms distinguish between customers with excellent and bad credit scores in order to reduce risk and improve profitability. Although numerous credit scoring techniques have been developed, our approach outperforms them and showed statistical improvement in terms of various performance measures on all Australian, German and Taiwan datasets. Our model SCMVE successfully utilizes self-organizing maps (SOM) and a soft voting-based ensemble classifier to create a hybrid model built for classification tasks of credit scoring. Multi-layer clustering of the dataset enhances the performance of our model by clustering similar data. The final classification is done using a weighted ensemble classifier. The weights are appropriately generated using the Sailfish Optimizer based on the predictive capabilities of the individual classifiers. The model gives an accuracy of 95.59% on the Australian dataset and 87% on the German credit dataset as well as 82.8% on the Taiwan credit dataset.

Future research can concentrate on improving the model's accuracy and efficiency through the selection of appropriate features. This can be achieved by incorporating data from various sources and evaluating the model's performance in real-life applications. It is important to prioritize the safety and fairness of the model to avoid discrimination and biases. By addressing these aspects, researchers can advance the field of machine learning and create more reliable and unbiased models.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability The datasets generated and/or analysed during the current study are available in https://archive.ics.uci.edu/dataset/143/statlog+australian+credit+approval,https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data and https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

Informed consent This article does not contain any studies with human participants or animals performed by any of the authors.

References

AghaeiRad A, Chen N, Ribeiro B (2017) Improve credit scoring using transfer of learned knowledge from self-organizing map. Neural Comput Appl 28(6):1329–1342

Bumacov V, Ashta A, Singh P (2014) The use of credit scoring in microfinance institutions and their outreach. Strateg Chang 23(7–8):401–413

Dastile X, Celik T, Potsane M (2020) Statistical and machine learning models in credit scoring: a systematic literature survey. Appl Soft Comput 91:106263

Dua D, Graff C (2017) UCI machine learning repository

Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugen 7(2):179–188

Gholamian M, Jahanpour S, Sadatrasoul S (2013) A new method for clustering in credit scoring problems. J Math Comput Sci 6:97–106

Guo S, He H, Huang X (2019) A multi-stage self-adaptive classifier ensemble model with application in credit scoring. IEEE Access 7:78549–78559

Hand DJ, Kelly MG (2002) Superscorecards. IMA J Manag Math 13(4):273–281

He H, Zhang W, Zhang S (2018) A novel ensemble method for credit scoring: adaption of different imbalance ratios. Expert Syst Appl 98:105–117

Henley WEM, Hand DJ (1996) Ak-nearest-neighbour classifier for assessing consumer credit risk. J Roy Stat Soc Ser D (Stat) 45(1):77–95

Hsieh N-C (2005) Hybrid mining approach in the design of credit scoring models. Expert Syst Appl 28(4):655–665

Hsieh N-C, Hung L-P (2010) A data-driven ensemble classifier for credit scoring analysis. Expert Syst Appl 37(1):534–545

Huang C, Li Y, Yao X (2019) A survey of automatic parameter tuning methods for metaheuristics. IEEE Trans Evol Comput 24(2):201–216

Kohonen T (1998) The self-organizing map. Neurocomputing 21(1–3):1–6

Kozodoi N, Lessmann S (2020) Multi-objective particle swarm optimization for feature selection in credit scoring. In: Workshop on mining data for financial applications. Springer, pp 68–76

Kuppili V, Tripathi D, Edla DR (2020) Credit score classification using spiking extreme learning machine. Comput Intell 36(2):402–426

Lappas PZ, Yannacopoulos AN (2021) A machine learning approach combining expert knowledge with genetic algorithms in feature selection for credit risk assessment. Appl Soft Comput 107:107391

Lau KW, Hujun Y, Simon H (2006) Kernel self-organising maps for classification. Neurocomputing 69(16–18):2033–2040

Lee T-S, Chiu C-C, Lu C-J, Chen I-F (2002) Credit scoring using the hybrid neural discriminant technique. Expert Syst Appl 23(3):245–254

Li X, Ying W, Tuo J, Li B, Liu W (2004) Applications of classification trees to consumer credit scoring methods in commercial banks. In: 2004 IEEE international conference on systems, man and cybernetics (IEEE Cat. No. 04CH37583), vol 5. IEEE, pp 4112–4117

Li S-T, Shiue W, Huang M-H (2006) The evaluation of consumer loans using support vector machines. Expert Syst Appl 30(4):772–782

Liu W, Fan H, Xia M (2022) Multi-grained and multi-layered gradient boosting decision tree for credit scoring

Martens D, De Backer M, Haesen R, Vanthienen J, Snoeck M, Baesens B (2007) Classification with ant colony optimization. IEEE Trans Evol Comput 11(5):651–665

Nalič J, Martinovič G, Žagar D (2020) New hybrid data mining model for credit scoring based on feature selection algorithm and ensemble classifiers. Adv Eng Inform 45:101130

Nassef MGA, Hussein TM, Mokhiamar O (2021) An adaptive variational mode decomposition based on sailfish optimization algorithm and gini index for fault identification in rolling bearings. Measurement 173:108514

Onan A (2018a) Biomedical text categorization based on ensemble pruning and optimized topic modelling. Comput Math Methods Med 2018(1):2497471
Onan A (2018) An ensemble scheme based on language function analysis and feature engineering for text genre classification. J Inf Sci 44(1):28–47
Onan A (2019a) Consensus clustering-based undersampling approach to imbalanced learning. Sci Program 2019(1):5901087
Onan A (2019b) Two-stage topic extraction model for bibliometric data analysis based on word embeddings and clustering. IEEE Access 7:145614–145633
Onan A (2021a) Sentiment analysis on massive open online course evaluations: a text mining and deep learning approach. Comput Appl Eng Educ 29(3):572–589
Onan A (2021b) Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks. Concurr Comput Pract Exp 33(23):e5909
Onan A (2022) Bidirectional convolutional recurrent neural network architecture with group-wise enhancement mechanism for text sentiment classification. J King Saud Univ Comput Inf Sci 34(5):2098–2117
Onan A (2023a) Gtr-ga: harnessing the power of graph-based neural networks and genetic algorithms for text augmentation. Expert Syst Appl 232:120908
Onan A (2023b) Srl-aco: a text augmentation framework based on semantic role labeling and ant colony optimization. J King Saud Univ Comput Inf Sci 35(7):101611
Onan A, Korukoğlu S, Bulut H (2016a) Ensemble of keyword extraction methods and classifiers in text classification. Expert Syst Appl 57:232–247
Onan A, Korukoğlu S, Bulut H (2016b) A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification. Expert Syst Appl 62:1–16
Onan A, Korukoğlu S, Bulut H (2017) A hybrid ensemble pruning approach based on consensus clustering and multi-objective evolutionary algorithm for sentiment classification. Inf Process Manag 53(4):814–833
Pławiak P, Abdar M, Acharya RU (2019) Application of new deep genetic cascade ensemble of svm classifiers to predict the Australian credit scoring. Appl Soft Comput 84:105740
Reichert AK, Cho C-C, Wagner GM (1983) An examination of the conceptual issues involved in developing credit-scoring models. J Bus Econ Stat 1(2):101–114
Runchi Z, Liguo X, Qin W (2023) An ensemble credit scoring model based on logistic regression with heterogeneous balancing and weighting effects. Expert Syst Appl 212:118732
Safi SA-D, Castillo PA, Faris H (2022) Cost-sensitive metaheuristic optimization-based neural network with ensemble learning for financial distress prediction. Appl Sci 12(14):6918
Şen D, Dönmez CÇ, Yıldırım UM (2020) A hybrid bi-level metaheuristic for credit scoring. Inf Syst Front 22(5):1009–1019
Shadravan S, Naji HR, Bardsiri VK (2019) The sailfish optimizer: a novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng Appl Artif Intell 80:20–34
Simumba N, Okami S, Kodaka A, Kohtake N (2022) Multiple objective metaheuristics for feature selection based on stakeholder requirements in credit scoring. Decis Support Syst 155:113714
Singh P (2017) Comparative study of individual and ensemble methods of classification for credit scoring. In: 2017 International conference on inventive computing and informatics (ICICI). IEEE, pp 968–972
Suleiman S, Ibrahim A, Usman D, Isah BY, Usman HM (2021) Improving credit scoring classification performance using self organizing map-based machine learning techniques. Eur J Adv Eng Technol 8(10):28–35
Transpire Online sailfish optimizer (sfo) (2019) A novel method motivated from the behavior of sailfish for optimal solution
Tripathi D, Edla DR, Cheruku R, Kuppili V (2019) A novel hybrid credit scoring model based on ensemble feature selection and multilayer ensemble classification. Comput Intell 35(2):371–394
Tripathi D, Edla DR, Kuppili V, Bablani A (2020a) Evolutionary extreme learning machine with novel activation function for credit scoring. Eng Appl Artif Intell 96:103980
Tripathi D, Edla DR, Kuppili V, Dharavath R (2020b) Binary bat algorithm and rbfn based hybrid credit scoring model. Multimedia Tools Appl 79(43):31889–31912
Van Gestel IT, Baesens B, Garcia IJ, Van Dijcke P (2003) A support vector machine approach to credit scoring. In: Forum Financier-Revue Bancaire et Financiaire Bank en Financiewezen, pp 73–82
West D (2000) Neural network credit scoring models. Comput Oper Res 27(11–12):1131–1152

Xia Y, Liu C, Da B, Xie F (2018) A novel heterogeneous ensemble credit scoring model based on bstacking approach. Expert Syst Appl 93:182–199
Xia Y, Zhao J, He L, Li Y, Niu M (2020) A novel tree-based dynamic heterogeneous ensemble method for credit scoring. Expert Syst Appl 159:113615
Xiao XJ, Zhong ZY, Xie L, Xin G, Liu D (2020) Cost-sensitive semi-supervised selective ensemble model for customer credit scoring. Knowl-Based Syst 189:105118
Zhang W, Yang D, Zhang S (2021a) A new hybrid ensemble model with voting-based outlier detection and balanced sampling for credit scoring. Expert Syst Appl 174:114744
Zhang W, Yang D, Zhang S, Ablanedo-Rosas JH, Xin W, Lou Yu (2021b) A novel multi-stage ensemble model with enhanced outlier adaptation for credit scoring. Expert Syst Appl 165:113872
Zhou Y, Shen L, Ballester L (2023) A two-stage credit scoring model based on random forest: evidence from Chinese small firms. Int Rev Financ Anal 89:102755

Authors and Affiliations

Indu Singh[1] · D. P. Kothari[2] · S. Aditya[1] · Mihir Rajora[1] · Charu Agarwal[3] · Vibhor Gautam[4]

✉ Indu Singh
indusingh@dtu.ac.in

D. P. Kothari
dpkvits@gmail.com

S. Aditya
s.aditya.me@gmail.com

Mihir Rajora
mihee20@gmail.com

Charu Agarwal
acharu848@gmail.com

Vibhor Gautam
vibhorgautam907@gmail.com

[1] Department of Computer Science and Engineering, Delhi Technological University, Delhi 110042, India

[2] Department of Electrical Engineering, Visvesvaraya National Institute of Technology, Nagpur, Maharashtra 440010, India

[3] Department of Electronics and Communication Engineering, National Institute of Technology, Hamirpur, Himachal Pradesh 177005, India

[4] Department of Electronics and communication Engineering, Delhi Technological University, Delhi 110042, India