



Review Article

Empirical Analysis of Ensemble Learning for Imbalanced Credit Scoring Datasets: A Systematic Review

Sudhansu R. Lenka ¹, Sukant Kishoro Bisoy,¹ Rojalina Priyadarshini,¹ and Mangal Sain ²

¹Department of CSE, C.V. Raman Global University, Bhubaneswar, India

²Division of Computer Engineering, Dongseo University, Busan 47011, Republic of Korea

Correspondence should be addressed to Mangal Sain; mangalsain1@gmail.com

Received 22 April 2022; Accepted 31 May 2022; Published 15 June 2022

Academic Editor: Kalidoss Rajakani

Copyright © 2022 Sudhansu R. Lenka et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Credit scoring analysis has gained tremendous importance for researchers and the financial industries around the globe. It helps the financial industries to grant credits or loans to each deserving applicant with zero or minimal risks. However, developing an accurate and effective credit scoring model is a challenging task due to class imbalance and the presence of some irrelevant features. Recent researches show that ensemble learning has achieved supremacy in this field. In this paper, we performed an extensive comparative analysis of ensemble algorithms to bring further improvements in the algorithm oversampling, and feature selection (FS) techniques are implemented. The relevant features are identified by utilizing three FS techniques, such as information gain (IG), principal component analysis (PCA), and genetic algorithm (GA). Additionally, a comparative performance analysis is performed using 5 base and 14 ensemble models on three credit scoring datasets. The experimental results exhibit that the GA-based FS technique and CatBoost algorithm perform significantly better than other models in terms of five metrics, i.e., accuracy (ACC), area under the curve (AUC), F1-score, Brier score (BS), and Kolmogorov-Smirnov (KS).

1. Introduction

Credit risk assessment is one of the most sensitive issues in the financial industry which identify the position of the potential borrower. Different types of risks are associated with the banking industries which may affect their business and their customers. Credit scoring is one of the major risks associated with the banks; it helps to make crucial decisions to lend some loan to the applicant or not. The banking sector access the creditworthiness of their applicants to grant loans by implementing the credit scoring models. Thus, developing an effective credit scoring model has become a demanding tool for researchers and the financial industries to precisely distinguish risky customers from nonrisky ones [1].

The credit scoring analysis is often treated as a binary classification problem, in that it determines whether the new credit applicants are “good” or “bad” by comparing their socioeconomic attributes. The models are initially developed using statistical methods, such as discriminant analysis and logistic regression (LR) [2], which are the most

common methods in this category. AI-based credit scoring models were proposed in recent decades to optimize accuracy and minimize error rates. Some commonly used machine learning (ML) techniques are decision tree (DT) [3], k-nearest neighbors (KNN) [4], support vector machine (SVM), and Naïve Bayes (NB) [5]. AI-based models gain more popularity than statistical models due to high accuracy labels [6], can easily handle nonlinear classification problems [7], and effectively handle high-dimensional datasets [6–8]. However, the credit scoring models are not always fully machine-dependent. In the semiautomated systems, before approving loans to the applicant, banks process the applications through two steps. First, it should be approved by the financial analyst (or experts), and then, the approved applications must be processed through computational models [9]. In these systems, the loans are approved based on the decisions made by the combined effect of expert knowledge and ML techniques. But, in the case of automated credit scoring systems, the applications are approved based on the decisions made by the ML techniques. However,

semiautomated credit scoring models are very rarely proposed in the literature [10, 11]. In this study, only the fully automated credit scoring models are discussed.

A single machine learning algorithm may not provide the best results in every case. In recent works [12, 13], ensemble models are developed for credit scoring problems, which results in more advanced and accurate models than single classification methods. The performance of the ensemble models improves by compensating the limitations of the base learners [14]. The performance can be further enhanced by implementing certain preprocessing mechanisms, such as feature selection (FS) and resampling the instances. Several studies [15, 16] have applied FS techniques in their credit scoring models to reduce the high-dimensional feature space and improve the overall performance of the model. In the paper [15], five traditional FS techniques, such as *t*-test, correlation matrix, stepwise regression, PCA, and factor analysis, are proposed to build a bankruptcy prediction model, and their performances are analyzed using MLP neural networks. In the research work of [16], multiple FS techniques are used to build the credit scoring model. FS techniques like LDA, rough set theory, DT, and F1-score were used, and the performances are examined through an SVM classifier. To improve the accuracy and stability of the credit scoring model, GA and ANN are used to select the optimal features [17].

The lack of a balanced dataset, i.e., a dataset with equally populated tuples for each class, creates a problem for the intended classifier. Skewness towards the positive class requires extra caution from the side of the model designer to make this impact as negligible as possible [18]. Imbalanced data is one of the common problems in credit scoring datasets, where the number of “bad” customers is much less than that of “good” customers. This makes the classifier biased towards majority class samples and leads to huge financial losses when the classifier incorrectly predicts the bad customer as good. Synthetic minority oversampling technique (SMOTE) is one of the widely used resampling techniques to deal with imbalanced datasets, and it achieves optimized performance by oversampling the minority class samples [19].

From the above studies, it is understood that the credit scoring models are designed considering three factors, i.e., (1) ensemble methods, (2) resampling, and (3) FS techniques. Most of the researchers implemented either one factor or a combination of any two factors to build the models. To the best of our knowledge, very few articles might have implemented all the three factors in their credit scoring models. Table 1 shows the studies related to credit scoring models, five papers have combined ensemble and resampling techniques, and four papers have combined FS and ensemble techniques. However, none of the papers have implemented all the three factors in their models. To fill this research gap, this paper proposed a credit scoring model by considering simultaneously all the three factors. In this paper, all three factors are considered in different phases to build an effective and accurate credit scoring model. In the resampling phase, SMOTE oversampling method was applied to tackle the imbalanced dataset. In the FS phase,

TABLE 1: Studies of credit scoring models.

Year	Paper	FS approach	Ensemble method	Resampling
2012	[21]		✓	
2021	[22]		✓	✓
2014	[23]		✓	
2015	[24]	✓	✓	
2015	[25]	✓		
2016	[26]	✓	✓	
2017	[27]		✓	
2017	[17]	✓	✓	
2018	[28]		✓	✓
2018	[29]		✓	
2018	[30]	✓		
2019	[31]	✓	✓	
2020	[32]		✓	
2020	[33]		✓	✓
2020	[34]		✓	✓
2021	[35]		✓	✓

three FS techniques, namely, IG, PCA, and GA, are employed to identify the informative features, which help to reduce the models’ dimensionality and complexity. In each phase, 19 base and ensemble of classifiers are used for model building. The baseline classification algorithms (i.e., LR, SVM, DT, NB, and KNN) and the ensemble of classifiers (i.e., bagging, boosting, and tree-based) are used in the experiment using three publicly available credit scoring datasets, i.e., Australian, German, and Japan. The predictive performances of the credit scoring models are evaluated against five evaluation metrics: ACC, AUC, F1-score, BS, and KS. Additionally, the ranks of each model are obtained using the Friedman and Nemenyi post hoc statistical tests [20]. The performances of the classifiers are compared by conducting a set of experiments in terms of the above metrics. In brief, the contributions of this paper are as follows:

- (1) It implements all three approaches: resampling, FS, and ensemble methods on three credit scoring datasets
- (2) It provides experimental results of 5 base classifiers and 14 ensembles of classifiers in terms of ACC, AUC, F1-score, BS, and KS
- (3) It implements three FS techniques, i.e., IG, PCA, and GA, to select the relevant features
- (4) It implements five sets of experiments to identify the best combination of machine learning algorithm and the FS technique that could able to build an accurate and reliable credit scoring model

The main objective of this work is to identify the best combination of FS technique and machine learning algorithms to build an accurate and reliable ensemble-based credit scoring model.

The remaining part of the paper is outlined as follows: Section 2 presents the literature survey of related works, ensemble, and feature selection techniques for the credit scoring model. In Section 3, the proposed methodology with preprocessing techniques is discussed. Experimental setup, evaluation metrics, and statistical tests are given in Section 4. In Section 5, results are discussed, and finally, we draw conclusions and future work in Section 6.

2. Literature Review

In recent decades, most credit scoring models have been proposed using an ensemble of classifiers due to their superior performance. In this section, the credit scoring and its related works, ensemble learning techniques in credit scoring, and credit scoring models with and without feature selection techniques are reviewed.

2.1. Credit Scoring and Its Related Works. The credit scoring model act as a decision-making system for the banks; it helps to make crucial decisions to approve a loan to the applicant or not. The models are designed using different methods, such as judgmental methods, statistical methods, rule-based methods, reject inference methods, profit-based methods, and machine learning methods. In the beginning, due to the nonavailability of data science methodology, the judgmental approaches of the expert team were being followed and approved the loans by reviewing the application form. In certain situations, accurately estimating the risk may be challenging for experienced professionals [24]. But, with the development of the technologies, many effective statistical credit scoring models have been proposed. Statistical techniques, such as LR and LDA have been proved to be superior credit scoring models as compared to the traditional expertise-based models [24]. These methods can determine the linear relationship between the attributes and the class variables. However, they cannot analyze the nonlinear mappings between the variables of the credit scoring data [36]. In rule-based credit scoring models, the rule extraction algorithms are combined with machine learning techniques to predict the creditworthiness of the applicants. The main advantages of these models are they can easily identify the patterns in complex problems and using these patterns the rules can be easily extracted. But, it is difficult to implement these rules in large dimensional credit scoring problems [9]. Traditionally, the credit scoring models are designed using the data that contains the records of only accepted applicants. Then, these models will have a selection bias, since they are trained only on accepted applicants and not on rejected applicants [37]. In reject inference credit scoring methods, the models are trained using labeled (accepted applicants) and unlabelled (rejected applicants) [38]; such models can correctly classify all types of loan applications. In the profit-based credit scoring methods, the model was aimed at maximizing the profit by granting loans to the applicants. These models gain profit by maximizing the benefits and minimizing the losses due to bad credits [39]. The credit scoring models designed using machine learning techniques become more popular. ML

methods can automatically extract the relevant information from the instances and can build advanced credit scoring models. In the last two decades, ML has achieved more popular and is effectively used to estimate the probability of defaulters. It can automatically extract the relevant information from the instances and can build advanced credit scoring models. In the studies [40], it has been observed that ML algorithms have achieved significantly better results than statistical methods. However, there are some limitations with these methods, such as (1) hyperparameter tuning is required, (2) normally stick at local minima, (3) it may overfit, and (4) computationally more expensive to train the model.

Different ML techniques have been employed to build the credit scoring models. SVM separates the class samples by an optimal hyperplane and thereby significantly increases the performance of the models [41]. A probabilistic-based Gaussian algorithm was proposed to build the credit scoring model that gives better accuracy than LR and SVM [42]. However, financial industries are not able to correctly identify the defaulters by employing a single classification algorithm. As a result, to mitigate the default risks, the researchers have proposed high predictive models by employing ensemble methods [34, 35].

In the study [43], a set of experiments was conducted and determined that an ensemble model performs better than a single classifier. In some ensemble methods, such as AdaBoost, gradient boosting decision tree (GBDT), and extreme gradient boosting (XGBoost), DTs are used as a base classifier [44, 45]. According to [46], the loan defaulters list can be estimated in a better way by employing RF than LR and KNN. Similarly, RF outperforms the other traditional classifiers, such as SVM, KNN, and LR for predicting the best borrowers in peer-to-peer lending [46]. RF is an ensemble algorithm that generates good accuracy and generates a model that can avoid overfitting, faster, and above all, effectively handle outliers and noises [27].

Credit scoring datasets may include some unimportant or redundant features that increase the training time and reduce the algorithms' performance level. FS technique helps to minimize the complexity, reduces the training time, and improves the accuracy level of the algorithms by selecting the informative features from the datasets [47]. Each FS technique has some pros and cons, like the filter method selects the features based on a certain ranking criterion. The top-ranked features are randomly used in the classification process while ignoring the impact of the features on the classifier's performance. The wrapper method selects the optimal feature subset according to the classifier's performance. This wrapper method results in the best feature subset, but computationally, it is more expensive. The hybridization method combines the application of both methods. The advantages of both methods are reflected in this hybrid approach; i.e., it achieves high classification accuracy and is computationally less expensive [48]. In the work of [49], two FS algorithms were proposed using a set of ML algorithms, such as LR, RF, SVM, MARS, XGBoost, and deep neural networks (DNN). XGBoost and DNN incorporating RF-based new approach (NAP) FS method result in high

ACC and AUC, respectively. In the paper [40], five traditional FS techniques, such as *t*-test, correlation matrix, step-wise regression, PCA, and factor analysis, are proposed to build a bankruptcy prediction model, and their performances are analyzed using MLP neural networks. In the research work of [15], multiple FS techniques are used to build the credit scoring model. FS techniques like LDA, rough set theory, DT, and F1-score were used, and the performances are examined through an SVM classifier.

2.2. Ensemble Learning Techniques in Credit Scoring. Improving the performance level is one of the biggest issues in ensemble models. Ensemble models can be implemented using single base learners with different variants (called a homogenous ensemble) or combining different base learners (called a heterogeneous ensemble). By the application of multiple algorithms, the ensemble model outcomes increase as compared to the outcomes of each base algorithm. It has been universally accepted that the diversity and the performance of the base learners are two key factors of ensemble models. To improve the generalization and robustness of the ensemble models, it is needed to focus on these two key factors. It has been observed that the diversified base learner enhances the performance of the ensemble model [27]. To create diversification, it is required to train the base learners using different data subsets. Bagging [50] and boosting [26, 51–53] are the two common approaches to generate diverse members. Building an ensemble model undergoes two stages, namely, the creation of diverse base learners and the combining of the output of the learners. The outputs can be combined using techniques such as majority voting, weighted average, performance weighting, and stacking [6, 47].

2.3. Credit Scoring Models with and without Feature Selection Techniques. Dataset may include some irrelevant or redundant features which may increase the complexity of the training process, and it leads to a reduction in the performance level of the model. The feature selection (FS) technique helps to reduce the complexity of the problem by eliminating the irrelevant features and also helps to increase the predictive capability of the model [15, 54]. In the paper [30], a hybrid FS technique has been proposed HMPGA, in which three feature subsets are shortlisted using three different filter methods, such as IG ratio, F1-score, and Pearson's correlation. Then, the optimal feature subset is finalized using a wrapper method called MPGA (multiple population genetic algorithm). Similarly, another hybrid FS method IGDFS for credit scoring problems has been proposed in [29]. It implements the IG filter method to select the feature subsets and the best subset is selected using GA. FS techniques help to build models using complicated nonlinear related variables without considering the model's assumptions [26]. It also helps to identify the relationships between independent and dependent variables in large datasets and reduces the training processing time of the models, especially for large datasets.

In the literature, several credit scoring models have been proposed without implementing FS techniques. A set of ML

algorithms, such as LR, Classification and regression trees (CART), ANN, and SVM have been implemented in these models. In the paper [55], a three-layered neural network model is proposed using a back-propagation learning algorithm to predict whether to grant or reject the loan application in an automated processing system. A hybrid credit scoring model was proposed in the paper [56] by combining genetic programming with deep learning network. Genetic programming was applied to extract rules and deep learning network was used to build the credit scoring model. In the paper [57], a few limitations in the credit scoring problems were identified, such as correctly setting the cut-offs for classifying good/bad borrowers, dealing with imbalanced credit datasets, and the implementation of ensemble methods. A profit-based credit scoring model using reinforcement learning is proposed [58] to determine the optimal threshold value. Similarly, to address the imbalance credit scoring problems, an improved SMOTE algorithm using AdaBoost and deep learning technique is proposed [35].

3. Proposed Methodology

In the proposed methodology, three methods are integrated, i.e., FS, resampling of minority class instances using SMOTE, and an ensemble of learners to build an effective credit scoring model. The flow of the proposed work is illustrated in Figure 1. It consists of four phases: (1) data preprocessing, (2) resampling using SMOTE, (3) feature selection, and (4) model generation. In the subsequent sections, the details of each phase are discussed.

3.1. Data Preprocessing. In the first phase, some necessary steps are employed to preprocess the dataset. Data preprocessing helps to enhance the accuracy and efficacy of the classifier. Each dataset undergoes different methods to make it more appropriate for the classification process. In this phase, data cleaning and normalization technique are implemented. In data cleaning, missing values are handled which is a common issue in many real-world problems. Data imputation is a method used to fill the missing values of an attribute by using the existing information. Mean imputation is a technique in which the mean of the existing data replaces the missing values.

In most cases, the domain range of the features differs by a large value. The features with a higher range of values create more influence on the classification model. Data normalization is another preprocessing technique in which the feature values are transformed into a particular range so that all the features will have equal influence. Normally, all the feature values are scaled within the range of [0, 1], which is expressed as

$$\text{Scaled}_x = \frac{\text{actual}_x - \min(\text{allvalues})}{\max(\text{allvalues}) - \min(\text{allvalues})}. \quad (1)$$

3.2. Resampling. This paper implements SMOTE to address the imbalanced issues before modeling. It oversamples the minority class subset by generating a specific number of artificial minority class samples. For each instance, x_i belongs to

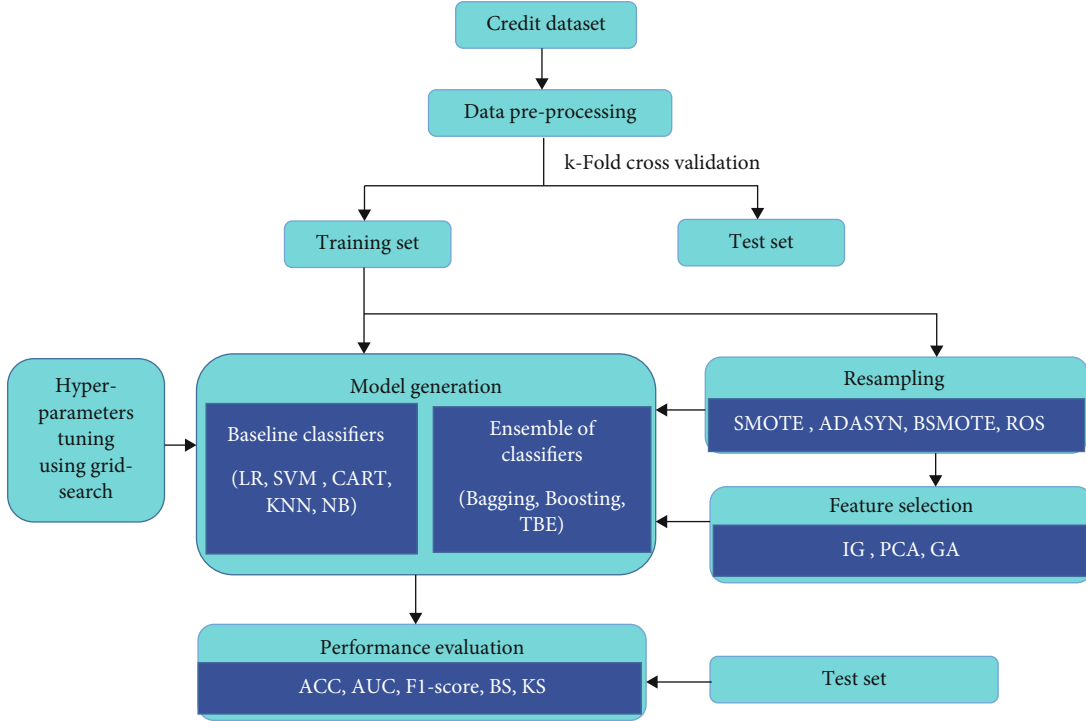


FIGURE 1: Flow diagram of the proposed model.

the minority class subset, T_{\min} , and its k -nearest neighbors from the minority class subset are identified using Euclidean distances. Then, a random sample x_j is selected from T_{\min} , and finally, along the line segment between x_i and x_j , a new minority sample x_{new} is obtained using the following equation:

$$x_{\text{new}} = x_i + (x_j - x_i) * \text{rand}(0, 1), \quad (2)$$

where $\text{rand}(0, 1)$ generates random numbers between 0 and 1. x_{new} is added to T_{\min} to make the imbalanced ratio (IR) to 1.

Additionally, three different oversampling methods are used in the experiment, namely, random oversampling (ROS), adaptive synthetic sampling (ADASYN) [59], and Borderline-SMOTE [60]. ROS randomly replicates the minority class samples to make its number equal to the number of majority class samples. ADASYN assigns different weights to the minority class samples, and more weights are assigned to the samples that are hard to classify. More synthetic samples are generated for instances having higher weights. But, in the case of Borderline-SMOTE, the samples that lie near the borderline are assumed to have more impact on classification. These samples are used to generate synthetic samples through interpolation.

3.3. Feature Selection. In the third stage, we applied three FS techniques, such as IG ratio, GA, and PCA to select the relevant features from the datasets. Through FS techniques, valuable feature subsets are chosen and help to achieve highly optimized model performance in the subsequent stages. FS technique not only improves the model efficiency but also reduces the complexity and running time of algo-

rithms. The three FS techniques are discussed in the following subsections.

3.4. Information Gain (IG). IG adopts feature ranking principles to find out the best features that are very much related to the class variables. The features with high IG are selected to enhance the classification of the model [54]. The IG of a feature is computed by evaluating the overall reduction in entropy. Entropy quantifies the expected value of a feature that is used while classifying an instance. Let X be an input feature vector and Y be the corresponding class variable; the entropy of Y is computed by taking the probability distribution of each $y \in Y$, which is expressed as

$$\text{Info}(Y) = - \sum_{y \in Y} P(y) * \log_2 P(y), \quad (3)$$

where $p(y)$ is the probability of Y belonging to class y . Now considering the feature vector X , the entropy is defined as

$$\text{Info}\left(\frac{Y}{X}\right) = - \sum_{x \in X} P(x) * \sum_{y \in Y} P\left(\frac{y}{x}\right) * \log_2 P\left(\frac{y}{x}\right). \quad (4)$$

Finally, the IG of feature vector X is defined as

$$\text{IG}(X) = \text{Info}(Y) - \text{Info}\left(\frac{Y}{X}\right). \quad (5)$$

3.5. Genetic Algorithm (GA). GA [61] is an evolutionary heuristic search algorithm where selection, cross-over,

and mutation operators are used to find the optimal feature subset. It is an optimal search technique in which the chromosome represents the feature subset in the form of binary strings. Each feature subset is evaluated based on the fitness score. The feature subset is selected based on a higher fitness score. The features are selected from the subset if the bit is 1; otherwise, the feature is discarded. According to the study [62], if a single metric is used to evaluate the performance of the classification algorithm, AUC is a more appropriate choice to be used in the fitness function to evaluate each individual of the population.

3.6. Principal Component Analysis (PCA). PCA is a feature transformation technique used to transform the high-dimensional feature vector \mathbb{R}^d to lower-dimensional relevant feature vector \mathbb{R}^p [63], where $d > p$. The resultant features are principal components, which are evaluated using the eigenvalue of the covariance matrix of the feature vector. These orthogonal principal components are used to identify the correlated essential features.

3.7. Model Generation. In the final stage, the models are generated by applying a set of base and ensemble of classifiers. In the subsequent sections, the details of each classifier are discussed.

3.7.1. Baseline Classifiers

- (1) Logistic regression (LR): LR is the most traditional statistical method and is widely used in credit scoring problems [43]. For the binary classification problems, it transforms the output from continuous values $[-\infty, +\infty]$ to 0 or 1. For credit scoring problems, LR can be used to estimate the probability of a customer's default using the logistic transformation function
- (2) Support vector machine (SVM): SVM classifies the instances of both the classes by an optimal hyperplane such that the data points of both the classes are separated by maximal distance [64]. SVM can also classify the nonlinear data points more accurately than other methods. It handles such classification by transforming the data points into high-dimensional space using different kernel functions, such as linear, polynomial, Gaussian, and radial basis functions (RBF)
- (3) Naïve Bayes (NB): NB is a probabilistic-based ML algorithm that uses the Bayes theorem and provides better predictive performance for high-dimensional input feature vectors [65]. It implements Bayes' rule and can predict whether the applicant is eligible for a loan or not. The rule estimates the probability that an instance x belongs to class y having the highest posterior probability
- (4) Decision tree (DT): DT is very popular and easy to interpret because its graphical structure looks very similar to human reasoning [66]. In DT, the attributes are represented as nodes, the branches split the instances into smaller subsets, and the terminal

nodes represent the class label. The tree is built using the training instances and the class label of each test sample can be easily predicted from its structure. This study employs a classification and regression tree (CART) to build the credit scoring model. It implements all the possible combinations for splitting each attribute to build an optimized model

- (5) K-nearest neighbor (KNN): KNN is a nonparametric classifier that does not need to train the model [4]. The classifier can predict the new instances based on k-nearest training instances. For each test observation, its k-nearest training samples are identified and the class outcome is predicted based on the majority class of k-nearest neighbors

3.7.2. Ensemble of Classifiers. Some of the widely used ensemble techniques are boosting, bagging, and tree-based ensembles.

- (1) Boosting: boosting operates sequentially by implementing a set of weak learners (normally shallow DT). Initially, a weak learner gets trained on the training set by assigning weights $(1/N)$ to all the N samples. In every iteration, the instances that were incorrectly classified in the current step are given more weight in the next iteration to correctly classify those misclassified instances [67]. All the instances with their respective weights are used to train the next weak classifier. More weightage is assigned to the classifier that performs well. By repeating the process, the performance of the classification increases by taking more weak learners. Finally, the resultant superior model is generated by linearly combining the classifiers with higher weights
- (2) Bagging: bagging is another ensemble approach in which different sets of training subsets are generated using the bootstrap aggregation method [50]. In this approach, K different classifiers are trained by randomly selecting the training subsets with replacement. Each classifier predicts the output for each input vector and the final output is generated using the majority voting technique. Random forest (RF) implements bagging approaches in which a set of DTs is constructed by selecting the training subsets using bootstrap samples and randomly selecting the features. At each round, different trees are built and each of them predicts the output for a given input pattern. The output predicted by each tree is aggregated and the final output is generated based on majority voting [68]
- (3) Tree-based ensembles (TBE): different tree-based ensemble methods, such as gradient boosting decision tree (GBDT), extreme gradient boosting (XGBoost) [69], light gradient boosting (LGBM) [70], and categorical boosting (CatBoost) [71] are used in this study. GBDT is a boosting algorithm in which a set of weak decision trees are combined to

build a strong ensemble model [67]. In this case, the sample data are updated along the negative gradient to reach a point where the algorithm converges globally [68]. XGBoost is an improved version of GBDT, which is designed to minimize the computational cost and above all increase the model efficiency both for classification and regression type problems. Like GBDT, XGBoost can handle overfitting problems by using the learning rate, number of boosting, the tree’s maximum depth, and subsampling [67]. As compared to GBDT, XGBoost increases the efficiency of the model by optimizing the objective function

The high performance of LGBM is due to the application of the “best-first” tree and histogram-based decision tree. XGBoost trained the model by increasing the size of the tree depth-wise, whereas LGBM adopts the “best-first” tree by growing the tree leaf-wise and limiting the depth-wise growth. The best-first helps to minimize the loss function rapidly but may lead to overfitting issues; LGBM prevents the tree depth and the splitting of the nodes.

CatBoost is another powerful GBDT-based algorithm that operates on two advanced algorithms, i.e., ordered boosting and techniques to handle categorical features. Through ordered gradient boosting techniques, it helps to reduce the biased gradient estimates and the overfitting issues. Categorical features are usually present in the credit scoring datasets. Normally, the hot-encoding technique is used to transform the categorical attributes into numeric values but it may lead to overfitting. CatBoost can easily handle the overfitting issue by converting these features to the gradient at each step.

4. Empirical Study

In this section, the experimental setup is designed to evaluate the performance of the proposed credit scoring models. It includes the descriptions of three credit scoring datasets and performance metrics to evaluate the experimental results, and finally, the performance of the classifiers is analyzed using statistical tests. The experiments are implemented in Python 3.8 on a Jupyter notebook on a PC with Intel Core 4 CPU, 4GB RAM, and Windows 10 operating system.

4.1. Credit Scoring Datasets and Experimental Settings. For the experimental analysis of different models, the three most commonly used credit scoring datasets were applied. These datasets are collected from the UCI ML repository¹, such as Australian, Japanese, and German datasets. Table 2 shows the brief descriptions of these datasets, and all these datasets are relatively small but contains the total data. The Australian credit dataset contains 690 instances, of which 307 are positive and the remaining 383 are negative, yielding an imbalance ratio (IR) of 1.25. Similarly, in the case of the Japanese dataset, the total number of samples is 690, with an imbalanced ratio (IR) of 1.25. The German dataset contains 1000 instances, out of which 700 are positive and 300 are

TABLE 2: Description of the credit scoring datasets.

Dataset	#instance	#feature	%Good	%Bad	IR
Australian	690	14	44.5	55.5	1.25
German	1000	20	70.0	30.0	1.25
Japanese	690	15	45.3	54.7	2.33

¹<https://www.ics.uci.edu/~mllearn/MLRepository.html>.

TABLE 3: Confusion matrix.

	Predicted bad (negative)	Predicted good (positive)
Actual bad (negative)	True negative (TN)	False positive (FP)
Actual good (positive)	False negative (FN)	True positive (TP)

negative, with a class distribution of 2.33. The “#instance” column represents the number of instances, the “#feature” column is the number of feature/attributes, “%Good” and “%Bad” column represents the percentage of good/positive and bad/negative applicants, and the IR column represents the imbalance ratio.

In the proposed work, an experiment is conducted to perform an extensive comparative analysis of the performances of baseline and ensemble models. In total, 5 base classifiers + 14 ensemble models = 19 models are used to perform comparative analyses on a set of credit score datasets. All the experiments have been conducted using 5-fold cross-validation to limit the effect of variability that occurs in random partitioning and able to achieve optimized results. Each dataset is partitioned into five folds, one fold is used as a test set to evaluate the model and the remaining four folds are used for training purposes. The experiments were carried out by repeating ten times the 5-fold cross-validation process for tuning the hyperparameters.

4.2. Evaluation Metrics. To evaluate the effectiveness of the credit scoring models, five evaluation metrics were used, i. e., accuracy (ACC), the area under the curve (AUC), the F1-score, Brier score (BS), and the Kolmogorov-Smirnov statistic (KS). These measures are most commonly used in this problem domain, as they cover each feature of the model’s performance. ACC, AUC, and F1-score metrics can be defined using the confusion matrix (shown in Table 3), in which one class is labeled as positive and the other class as negative. True negative (TN) and true positive (TP) represent the number of correctly classified negative and positive cases, respectively. Similarly, false negative (FN) and false positive (FP) represent the number of incorrectly classified negative and positive cases, respectively.

The ACC metric is used to find the proportion of instances that are correctly predicted by the model, which is defined in Equation (6). The AUC evaluates the discriminatory ability of the model based on the receiver operating characteristic curve (ROC). The AUC value lies between 0 (indiscernible) and 1 (perfectly discernible), and 0.5 indicates the predictive ability of a random classifier. The F1-score evaluates both positive and negative accuracies of the test samples by taking the weighted average of the precision

TABLE 4: Search space of the hyperparameters and the best parameters in each dataset.

Classifier	Parameter search space	Best parameters in German dataset	Best parameters in Australian dataset	Best parameters in Japan dataset
LR	$C \in [-15,15]$, solver \in [newton-cg, lbfgs, liblinear], penalty \in [l1, l2]	$C=4$, penalty=l2, solver=liblinear	$C=2$, penalty=l1, solver=liblinear	$C=10$, penalty=l2, solver=newton-cg
KNN	n_neighbors \in [2, 21], weights \in [uniform, distance], metric \in [euclidean, manhattan, minkowski]	Metric= manhattan, n_neighbors=9, weights=distance	Metric= manhattan, n_neighbors=11, weights=distance	Metric=manhattan, n_neighbors=15, weights=distance
SVM	$C \in [-10,10]$, gamma \in [-10,10], kernel \in [poly, rbf, sigmoid]	$C=2$, gamma=1, kernel = rbf	$C= 0.1$, gamma=1, kernel = rbf	$C=2$, gamma=1, kernel = rbf
CART	Criterion \in [gini, entropy], max_depth \in [1, 20], min_samples_leaf \in [1, 10]	Criterion=gini, max_depth=20, min_samples_leaf =10	Criterion=gini, max_depth=3, min_samples_leaf =20	Criterion=gini, max_depth=10, min_samples_leaf = 50
AdaBoost	learning_rate \in [0.0001,1], n_estimators \in [5,100]	learning_rate = 0.001, n_estimators = 50	learning_rate = 0.001, n_estimators =10	learning_rate = 0.1, n_estimators = 5
Bagging	n_estimators \in [10,1000]	n_estimators =1000	n_estimators =200	n_estimators =200
RF	bootstrap [True], max_depth \in [2, 50], max_features \in [2, 10], min_samples_leaf=[3,4,5], min_samples_split= [8, 10, 12], n_estimators \in [2,500]	Bootstrap= True, max_depth=20, max_features=2, min_samples_leaf= 3, min_samples_split= 8, n_estimators=300	Bootstrap= True, max_depth= 50, max_features= 3, min_samples_leaf=4, min_samples_split=10, n_estimators=100	Bootstrap= True, max_depth=20, max_features= 3, min_samples_leaf= 3, min_samples_split= 8, n_estimators=100
XGBoost	gamma = [0,0.1,0.2,0.4,0.8,1], learning_rate = [0.01, 0.1, 0.2, 0.3, 0.5, 0.6, 0.7], max_depth = [5,6,7,8,9,10], n_estimators = [50,65,80,100,200]	gamma= 0, learning_rate= 0.5, max_depth=10, n_estimators=200}	gamma=0.8, learning_rate=0.1, max_depth= 6, n_estimators= 65	gamma= 0.8, learning_rate= 0.3, max_depth=7, n_estimators=50
LGBM	learning_rate \in [0.1, 1], n_estimators \in [1, 200], max_depth \in [1, 10] min_child_weight \in [0.001, 1000] num_leaves \in [6, 50]	learning_rate=0.5, max_depth=4, min_child_weight= 0.01, n_estimators=200, num_leaves=18	learning_rate=0.3, max_depth= 1, min_child_weight= 0.01, n_estimators=50, num_leaves=36	learning_rate=0.5, max_depth=4, min_child_weight= 0.01, n_estimators=200, num_leaves=18
GBDT	learning_rate \in [0.1, 1], n_estimators \in [1, 200], max_depth \in [1, 10]	learning_rate= 0.7, max_depth=1, n_estimators=170}	learning_rate= 0.01, max_depth=1, n_estimators=10}	learning_rate=0.01, max_depth=1, n_estimators=10
CTB	learning_rate \in [0.1, 1], n_estimators \in [1, 200], max_depth \in [1, 10]	learning_rate=0.2, max_depth= 5, n_estimators=100	learning_rate=0.4, max_depth=4, n_estimators=200	learning_rate=0.2, max_depth=3, n_estimators=200

and recall scores. Precision defines the exactness, i.e. the number of samples that are predicted as positive that are actually positive, whereas recall defines the completeness, i.e. the number of positive samples that are correctly predicted.

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \quad (6)$$

$$\text{F1 - score} = \frac{2 \times \text{Precision} \times \text{Recall}}{(\text{Precision} + \text{Recall})}, \quad (7)$$

where

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})}, \quad (8)$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})}. \quad (9)$$

The BS metric evaluates the accuracy of the probability predictions. It computes the mean-squared error between the probability predictions and the actual label (0 or 1). It is defined as

$$\text{BS} = \frac{1}{N} \sum_{i=1}^N (P_i - y_i)^2, \quad (10)$$

where P_i is the predicted score and y_i is the actual label of the sample i^{th} sample.

The KS statistic is used to evaluate the maximum difference between the cumulative score of positive and negative samples.

TABLE 5: Performance of each classifier without resampling and FS.

Dataset	Metrics	LR	KNN	CART	NB	SVM	A-DT	A-SVM	A-NB	A-LR	XGB	LGB	GB	CTB	RF	B-SVM	B-NB	B-DT	B-LR	B-KNN
GER		76	74.2	69.9	72.4	76.2	68.1	69.8	65.6	75.2	76.5	76.8	77	77.6	76.5	76	71.8	76.3	75.2	75
AUS	ACC	83.3	84.1	82.7	79.5	83.7	82.6	71.1	60.6	82.3	87.1	86.3	86.5	87.5	87.3	85.6	80	86.7	82.5	84
JAP		82	86	80	80	85.5	80.5	76.1	52.5	84.3	88	87.2	87.8	88.6	87.9	85.6	80.3	87.1	87.1	85.8
	R_ACC	10.8	9.6	14.8	15.8	9.3	15	17.6	19	12.5	3.2	4.6	3.6	1	3.1	8.5	15.3	5.5	10.3	10
GER		73.2	72.4	63.8	76.9	62.2	63.2	66.3	73.5	78.7	78.8	78.9	79.3	81	79.9	79.5	77.1	78.6	78.5	74.2
AUS	AUC	90.2	90.7	81.9	89.7	93.1	82.4	88.5	75.1	90.1	93.1	93.1	93.7	94.1	93.4	93.1	90.3	93.4	93.2	91.4
JAP		90.3	90.5	79.4	88.9	92.8	80.6	84.7	76.1	88.8	93.3	93	86.4	94.1	93.5	92.6	90	93.3	92.7	91
	R_AUC	12.6	12	17.6	13	10.8	17.3	16	17	11.6	5.6	5.8	7	1	2.5	6.1	11.3	5	7	10.3
GER		53.7	46	49.1	57.8	49.9	47.9	2.1	1.8	54.2	55.5	56.4	56	56.5	51.3	49.8	58.5	55.6	53.6	45.6
AUS	F1	85.4	81.7	80.5	74.7	85.2	80.5	55.5	53.3	85.6	85.7	84.6	85.1	86	85.2	85.1	74.9	85.7	85.4	81.9
JAP		86.1	83.9	77.8	74.6	85.1	78.1	66	56.7	85.8	86.8	85.5	86.4	87.2	86.1	85.1	75.1	85.6	86.1	83.7
	R_F1	6.5	13.6	14.5	12	10	14.5	18	19	6.3	3.8	8	5.8	1.6	7.8	11	11	5.5	6.8	14
GER		0.17	0.18	0.29	0.227	0.2	0.29	0.188	0.2	0.24	0.17	0.18	0.16	0.16	0.16	0.2	0.22	0.17	0.16	0.17
AUS	BS	0.09	0.11	0.17	0.159	0.2	0.18	0.22	0.22	0.24	0.1	0.11	0.24	0.09	0.1	0.147	0.15	0.1	0.09	0.11
JAP		0.13	0.15	0.2	0.237	0.13	0.19	0.21	0.26	0.24	0.13	0.14	0.13	0.1	0.12	0.143	0.22	0.13	0.13	0.15
	R_BS	4.8	10.3	15.3	15	12	15	14.3	16.3	17.6	4.5	8.6	9.1	1	3	10.6	14	5.3	4.1	8.6
GER		44.9	29.6	34.1	35.2	21.5	33.8	32.6	30.5	41.9	41.3	38.8	45.2	38.6	39.9	40.3	36.2	36.9	44.4	35.8
AUS	KS	79.2	73.5	65.4	70.4	74.7	63.2	65.4	48.4	79.1	76.3	75.9	74.5	79.8	76.9	75	69.4	76.8	79.1	74.6
JAP		73.5	74.6	60.6	69.9	75.7	66.4	60.5	37.6	73.5	72.3	69.8	72.3	76.8	77.9	75.7	69.9	72.2	73.4	74.6
	R_KS	3.8	12.2	15.8	13.5	10.8	16.3	16.8	18.3	5	7.5	10.3	7.8	4	4.3	6.1	13.1	9.3	5.1	9.5

4.3. Statistical Tests. Even though we are using several measures but without statistical tests, we cannot conclude the model's superiority over others [20]. The statistical test needs to be carried out to show that the model's performance is statistically different from others. In this experiment, nonparametric tests are used to compare the performance of all the algorithms over the different datasets. The algorithms are ranked using the Friedman test, ranked 1 is assigned to the best algorithm, ranked 2 to the second-best, and so on. The test is carried out over each algorithm K using chi-square with $K-1$ degree of freedom. This value is computed using the rank (r_i^j) of each classifier j on each dataset $i \rightarrow 1 \dots N$, where N indicates the number of datasets. It is defined using the following equation:

$$X_F^2 = \frac{12N}{K(K+1)} \left[\sum_j \left(\frac{1}{N} \sum_i r_i^j \right)^2 - \frac{K[k+1]^2}{4} \right]. \quad (11)$$

The Friedman test is applied to determine the significant differences in performances of all the classifiers. If the null hypothesis that there is no significant difference is rejected, then the Nemenyi post hoc test can be applied to determine significant differences in performances between each pair of classifiers. According to the Nemenyi post hoc test [20], the performances of two or more classifiers are significantly different if their respective average ranks differ by at least the critical difference (CD), which is defined as

$$CD = q_{\alpha, \infty, L} \sqrt{\frac{K(K+1)}{12N}}, \quad (12)$$

where the value $q_{\alpha, \infty, L}$ is based on the studentized range statistic table.

4.4. Hyperparameter Tuning. The performance of the classifiers depends on the hyperparameters. To improve the performances significantly, these hyperparameters need to be modified. Therefore, the grid search method, a popular hyperparameter optimization method, is employed to determine the best parameters from a prespecified parameter list. Table 4 presents the parameter searching space for all the classifiers and the best parameter obtained for all the datasets. The classifiers used in this study are LR, SVM, KNN, CART, AdaBoost, bagging, RF, XGBoost, LGBM, GBDT, and CatBoost; all these classifiers have a set of hyperparameters that needs to be optimized. However, NB is the classifier whose classification is only based on the prior probability of the features in the training set to estimate the posterior probability, so no parameter tuning is essential for it.

5. Results and Discussion

In the experiment, we build 5 baseline classifiers, i.e., LR, SVM, DT, NB, and KNN, and 14 ensemble models are generated by combining the baseline classifiers with AdaBoost, bagging, and tree-based ensemble methods. In the AdaBoost method, 4 ensemble models are generated by using LR,

TABLE 6: Results of the oversampling methods using DT (bold indicate best results).

Dataset		SMOTE	ROS	ADASYN	BSMOTE
German	ACC	77.7	68.8	74.9	75.3
	AUC	77.0	69.5	75.1	74.8
	F1-score	77.9	68.7	75	75.3
	BS	0.296	0.322	0.33	0.34
	KS	60.87	64.2	63.1	59.7
Australian	ACC	83.8	81.8	81.6	81.5
	AUC	82.7	82.7	81.4	81.3
	F1-score	83.6	80.3	81.6	80.8
	BS	0.169	0.183	0.222	0.188
	KS	69.7	61.7	56.7	66.8
Japan	ACC	82.3	79.5	80.1	80.5
	AUC	81.7	80	80.3	82.3
	F1-score	82.6	80.9	81.4	82.9
	BS	0.198	0.188	0.183	0.178
	KS	57.3	62.8	60.6	59.9

SVM, DT, and NB algorithms as base classifiers, represented as A_LR, A_SVM, A_DT, and A_NB, respectively. Similarly, in the bagging approach, 6 ensemble models are generated, i.e., random forest, and the remaining 5 are generated by using DT, SVM, LR, NB, and KNN as base classifiers, which are represented as B_DT, B_SVM, B_LR, B_NB, and B_KNN. In the tree-based approach, 4 ensemble models are generated by using XGBoost (XGB), GBDT (GB), CatBoost (CTB), and LGBM (LGB) as base classifiers.

All the models are evaluated concerning ACC, AUC, and F1-score metrics in four separate experiments. They are as follows:

- (i) Performance analysis of each classifier without resampling and FS
- (ii) Performance analysis of each classifier with resampling but without FS
- (iii) Performance analysis of each FS technique
- (iv) Performance analysis of each classifier with resampling and GA-based FS technique

Finally, in the last two experiments, the performance analysis of each classifier using nonparametric statistical tests and the comparison of the computational cost of each classifier is performed.

5.1. Experiment I: Performance Analysis of each Classifier without Resampling and FS. In this experiment, the performance of all the five individual classifiers and 14 ensemble learnings is compared prior to oversampling and FS techniques to the datasets. Table 5 represents the results of each base and ensemble model. The best classifiers in each metric are highlighted in bold fonts. Additionally, we also rank the classifiers from best (rank-1) performer to worst performer

TABLE 7: Performance of each classifier before oversampling (BO) and after oversampling (AO) and bold font indicates when performance level increases after oversampling.

Classifier	ACC (BO)	ACC (AO)	AUC (BO)	AUC (AO)	F1-score (BO)	F1-score (AO)	BS (BO)	BS (AO)	KS (BO)	KS (AO)
LR	80.43	80.83	84.57	84.93	75.07	80.57	0.13	0.12	65.87	66.7
KNN	81.43	82.50	84.53	88.27	70.53	81.83	0.15	0.17	59.23	61.3
CART	77.53	80.77	75.03	80.30	69.13	79.57	0.22	0.21	53.37	54.3
NB	77.30	77.50	85.17	85.40	69.03	76.70	0.21	0.20	58.50	59.2
SVM	81.80	82.27	82.70	82.93	73.40	81.97	0.18	0.18	57.30	58.3
A-DT	77.07	80.23	75.40	80.47	68.83	79.67	0.22	0.25	54.47	54.2
A-SVM	72.33	74.67	79.83	82.43	41.20	71.67	0.21	0.20	52.83	52.63
A-NB	59.57	57.17	74.90	73.83	37.27	41.63	0.23	0.20	38.83	39.4
A-LR	80.60	81.67	85.87	86.10	75.20	82.13	0.24	0.24	64.83	65.8
XGB	83.87	85.43	88.40	90.53	76.00	85.77	0.13	0.13	63.30	64.3
LGB	83.43	85.87	88.33	92.70	75.50	80.97	0.14	0.15	61.50	62.8
GB	83.77	84.90	86.47	90.40	75.83	85.17	0.18	0.14	64.00	64.5
CTB	84.57	86.73	89.73	92.40	76.57	85.50	0.12	0.11	65.07	66.9
RF	83.90	86.33	88.93	90.27	74.20	86.80	0.13	0.13	64.90	63.5
B-SVM	82.40	82.30	88.40	89.57	73.33	81.03	0.16	0.18	63.67	63.98
B-NB	77.37	77.57	85.80	86.07	69.50	76.60	0.20	0.21	58.50	61.4
B-DT	83.37	84.37	88.43	88.63	75.63	85.63	0.13	0.14	61.97	62.8
B-LR	81.60	81.70	88.13	88.07	75.03	81.43	0.13	0.14	65.63	66.8
B-KNN	81.60	83.30	85.53	89.03	70.40	83.90	0.14	0.16	61.67	63.5

using Friedman's rank in terms of ACC (R_ACC), AUC (R_AUC), F1-score (R_F1), BS (R_BS), and KS (R_KS) for all the datasets. All the tests have been carried out by taking the level of significance (α) value equal to 0.05.

From Table 5, we can observe that CTB is the best performer and RF is the second best considering all the three datasets. The mean scores of CTB in terms of different metrics are ACC (84.56%), AUC (89.73%), F1 (76.56%), BS (0.116), and KS (65.06%). Among the base learners, LR is the best performer with a mean rank of 7.7 and CART is the worst performer with a mean rank of 15.6.

After applying ensemble methods, some improvements in performance levels were observed in most of the datasets for all the base learners. Especially, DT (CART) has shown maximum improvements as compared to other base learners. The accuracy level of DT has enhanced by 5.5% to 9.8% in all the datasets after applying bagging (RF). Similarly, XGBoost brings improvements in accuracy by 5.3% to 10%. In general, ensemble methods obtain significant improvements in different metrics as compared to all the base learners.

5.2. Experiment II: Performance Analysis of Each Classifier with Resampling but without FS Technique. Through this experiment, we aim to show the effect of resampling on the performance of the classifiers. In the 5-fold cross-validation process, oversampling is implemented 5 times on all the training folds, while in all the testing folds, the class distribution is kept intact. That is, all the classification algorithms are tested on the dataset having the original class distribution.

In this experiment, four popular oversampling methods are implemented on all three datasets using DT classification. Oversampling methods used in the experiment are random oversampling (ROS), ADASYN, SMOTE, and Borderline-SMOTE (BSMOTE). After implementing oversampling methods, the performances of DTs are measured in terms of ACC, AUC, F1-score, BS, and KS which are shown in Table 6.

Now to show the effect of oversampling on the classification algorithms, we balance all the datasets using SMOTE and the algorithms are trained on those balanced datasets. Table 7 illustrates the performances of all the base and ensemble learners before oversampling (BO) and after oversampling (AO) in terms of each metric. The bold fonts indicate the performance level of the classifier increase after implementing oversampling. After implementing SMOTE, it has been observed that most of the classifiers have shown improvements in ACC (i.e., 0.2-5%), AUC (i.e., 0.5-5%), F1-score (7-20%), and KS (i.e., 0.5-3%). But, very few classifiers have shown slight improvements in BS. From this experiment, we conclude that SMOTE method brings significant improvement in ACC, AUC, and F1-score metrics for most of the classifiers.

5.3. Experiment III: Performance Analysis of Each FS Technique. Through this experiment, we aim to study the impact of FS techniques on the performance of the classification algorithms. As stated above, the class level distribution of the training dataset is made balanced using SMOTE, and then, FS techniques are applied to these balanced datasets. Next, the top-ranked features are identified and the

TABLE 8: Performance of SVM classifier using IG FS technique (bold indicates best results).

Model	No. of FS	German					Australian					Japan				
		ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS
1	4	61.3	66.9	67.4	0.20	36.5	83.5	85.3	83.8	0.20	72.3	81.7	87.6	86.7	0.14	73.5
2	6	68	68.5	75.1	0.19	38.1	85.7	86.6	87.1	0.2	72.1	82.5	88.2	86.8	0.13	75.7
3	8	68.8	69.5	75.6	0.19	38.4	83.8	84.6	82.9	0.20	73.5	81.6	88.5	85.2	0.14	73.6
4	12	72.8	71.6	78.3	0.18	40.9	86.4	87.1	88	0.19	74.6	82.1	86.5	86	0.15	74.5
5	15	70.8	69.4	78.1	0.202	40.1	—	—	—	—	—	81.6	85.5	81.9	0.15	74.2
Mean		68.34	69.18	74.9	0.196	38.8	85.1	85.9	85.5	0.20	73.1	81.9	87.2	85.3	0.14	74.3

TABLE 9: Performance of SVM classifier using PCA (bold indicates best results).

Model	No. of components	German					Australian					Japan				
		ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS
1	4	70.8	71.1	77.5	0.19	46.7	84.1	84.2	81.6	0.13	70.7	75.8	74.6	70.2	0.16	55.6
2	6	73.2	72.3	79.8	0.19	38.5	86.5	86.6	82.3	0.12	69.7	74.9	73.8	69.4	0.17	56.6
3	8	74.4	74.1	80.7	0.17	46.9	81.6	82.11	79.3	0.13	68.8	75.4	74.4	70.2	0.17	55.8
4	12	70.8	70.6	77.6	0.18	45.14	86.95	88.33	86.01	0.112	73.37	83.09	83.76	82.05	0.136	68.72
5	15	69.6	68.7	77.3	0.18	45.95	—	—	—	—	—	82.6	83.2	81.44	0.138	67.35
Mean		71.76	71.3	78.5	0.18	44.63	84.76	85.29	82.29	0.122	70.62	78.35	77.95	74.66	0.156	60.82

classifier gets trained on these selected features. Finally, the testing set with the same set of features is used to evaluate the performance of the classification. The FS techniques are implemented and their parameters are adjusted according to the performance of the SVM classifier.

In the IG FS approach, only the number of selected features in each subset varies. Each feature subset is evaluated using ACC, AUC, F1-score, BS, and KS. The results of each subset are shown in Table 8. In the German and Australian datasets, model-4 (i.e., selecting 12 important features) obtains best results than others in the classification process. In the Japan dataset, model-2 (i.e., selecting 6 important features) obtains best results than others.

Similarly, in the case of PCA, the feature space is reduced by taking a different number of components. The best feature subset is determined based on SVM classification performance. The results of the classification are shown in Table 9. According to the classification report, model-3 (i.e., 8 numbers of components) obtains the best results in the German dataset and model-4 (i.e., 12 numbers of components) obtains the best results in the Australian and Japan dataset.

In the GA FS technique, a set of parameters with different combinations are chosen by referring to the studies. To obtain the best feature subset, a series of experiments are conducted with various ranges of population size [50-300], mutation rate [0.001-0.3], cross-over rate [0.01-0.9], and the number of generations [20-100]. Finally, Table 10 presents, the best combination of parameters that are obtained after comparing all the combinations of parameters. The features that are selected using these optimal parameters are the best feature set, which is then applied to the SVM classification algorithm; the results are shown in Table 11.

The features that are selected using IG and GA for all three datasets are shown in Table 12. In Australian and Jap-

TABLE 10: Optimized GA parameters.

Parameter	Values
Population size	50-300
Cross-over rate	0.5
Mutation rate	0.05
No. of generations	20-50
Fitness function	AUC
Stopping criteria	Maximum number of generations

anese datasets, the features are named as [X1-X14] and [F1-F15], respectively. Finally, Table 13 shows the mean values of each metric of all the FS techniques. From Table 13, we conclude that GA is the best FS technique, and the features that are selected using this technique are applied to all the datasets in the model generation phase.

5.4. Experiment IV: Performance Analysis of Each Classifier with Resampling and GA-Based FS Technique. In this experiment, the optimal feature subset of each dataset is employed by all the classification algorithms on the balanced training set. Table 14 presents the results of each algorithm after implementing the GA-based FS technique on the oversampled dataset. From Table 14, it is clear that the CTB and XGB are the two best classification algorithms of the credit scoring problems. The mean ACC, AUC, F1-score, BS, and KS values of CTB are 87.1%, 91.5%, 86.53%, 0.112, and 68.22%; and in case of XGB, the respective values of the corresponding metrics are 85.67%, 90.30%, 84.0%, 0.121, and 64.94%. From experiments I-III, it is clear that the performance level of most of the classifiers improves after employing SMOTE and the performance level gets further improved after implementing the GA-based FS technique on the oversampled dataset.

TABLE 11: Performance of SVM classifier using GA FS technique (bold indicates best results).

Model	German					Australian					Japan				
	ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS	ACC	AUC	F1-score	BS	KS
1	66.3	69.5	73.05	0.144	72.6	83.97	83.2	85.21	0.18	72.6	82.08	86.18	83.05	0.138	72.2
2	69.3	70.8	76.29	0.135	74.8	84.1	84.5	85.5	0.179	73.4	83.49	87.13	83.45	0.134	74.9
3	70.3	71.9	77.69	0.123	76.5	85.8	88	86.5	0.166	75.8	84.6	88.8	84.1	0.122	76.5
Mean	68.7	70.7	75.67	0.134	74.6	84.62	85.23	85.74	0.174	73.9	83.39	87.37	83.53	0.131	74.5

TABLE 12: Feature selected/features ordered rank-wise using GA and IG.

FS technique	Dataset	FS selected/feature order rank-wise
GA	German	Features selected: status of existing checking account, duration in months, credit history, credit amount, savings account/bonds, present employment since, installment rate in percentage of disposable income, personal status and sex, other debtors/guarantors, property, age, other installment plans, housing, number of existing credits at this bank, number of people being liable to provide maintenance for, and telephone
Info-gain	German	Feature order rank-wise: credit amount, status of existing checking account, duration in months, age in years, credit history, savings account/bonds, purpose, property, present employment since, housing, other installment plans, personal status and sex, foreign worker, other debtors/guarantors, instalment rate in percentage of disposable income, number of existing credits at this bank, job, telephone, present residence since, and number of people being liable to provide maintenance for
GA	Australian	Features selected: X2, X3, X4, X5, X7, X8, X9, X11, X12, X13, and X14
Info-gain	Australian	Feature order rank-wise: X8, X10, X14, X5, X7, X9, X13, X6, X4, X3, X2, X1, X11, and X12
GA	Japan	Features selected: F1, F4, F6, F9, F11, F13, and F15
Info-gain	Japan	Feature order rank-wise: F9, F11, F10, F15, F8, F4, F6, F3, F14, F13, F5, F2, F1, F7, and F12

TABLE 13: Comparison of mean results of all the FS techniques.

	IG	PCA	GA
Mean ACC	78.45	78.29	78.89
Mean AUC	80.76	78.18	81.12
Mean F1	81.90	78.48	81.65
Mean BS	0.18	0.15	0.15
Mean KS	62.07	58.69	74.36

5.5. *Experiment V: Performance Analysis of Each Classifier Using Statistical Tests.* In each experiment, the Friedman test is conducted to rank the classifiers according to their predictive performances. In all three cases, the p value is less than α (0.05). This indicates that the performance measures of the classifiers are significantly different, and therefore, the null hypothesis is rejected.

Finally, to make further analysis, a Nemenyi post hoc test [20] with $\alpha = 0.05$ is applied to make pairwise comparisons of each classifier using the mean ranks of all the classification algorithms. Table 15 presents the ranks of each classifier in terms of ACC (R_ACC), AUC (R_AUC), F1-score (R_F1), BS (R_BS), and KS (R_KS) and the last column presents the mean ranks. Figure 2 represents the graphical representation of the global ranks. The CD defines the mean-ranking score difference among the classifiers [20]. The algorithms present on the right-hand side of the diagram are considered the top-ranked algorithms, and among them, CTB is the best one. The graphical representation shows that CTB is the superior algorithm among all the base and ensemble models.

5.6. *Comparison of Computational Cost.* Computational cost is another important metric that needs to be considered for the credit scoring model. An ideal credit scoring model should respond quickly to whether to allocate loans to the applicants or not. Except for SVM, each base learner comparatively takes less training time than that of ensemble models, because the base learners get trained only once, while the ensemble models get trained multiple times. Moreover, tree-based ensemble methods, such as GBDT, XGBoost, LGBM, and CatBoost, take comparatively low computational time due to GPU computing systems. The computational cost of the classifiers is determined by computing the single training time [12], which represents the training time of a single cross-validation process. Figure 3 shows the average computational time of base and ensemble models. SVM consumes maximum computational time, so it may not be appropriate to design a credit scoring model using SVM in the CPU computing system. However, the computational cost of the ensemble model can be further improved by implementing advanced computing mechanisms, such as distributed environment and GPU computing systems. Hence, instead of more computational cost, ensemble models should be used to build the credit scoring models.

6. Conclusions and Future Work

Effectively analyzing the default customers is an important process to improve the financial status of banks and financial industries. Advanced credit scoring models are an effective tool needed to identify the default customers. As stated in the literature section, different approaches have been

TABLE 14: Performance of each classifier with resampling and GA-based FS technique.

Classifier	ACC			AUC			F1			BS			BSC			KS				
	German	Australian	Japan	R ₊	German	Australian	Japan	R ₊	German	Australian	Japan	R ₊	German	Australian	Japan	R ₊	German	Australian	Japan	
LR	75.2	84.3	85.5	10.1	80.2	88.9	87.6	11.6	74.5	83.6	84.1	11.8	0.15	0.092	0.111	2.1	43.01	79.9	74.2	5.5
KNN	79.3	86.5	82.2	8.6	89.1	87.6	88.6	11.3	81.2	84.1	81	8.3	0.18	0.112	0.135	10	33.5	75.4	75.7	12
CART	78.3	84.6	81.9	12.6	77.3	88.6	84.2	15.3	78.5	82.5	78.8	14	0.25	0.163	0.174	15.3	44.2	66.9	62.3	12.3
NB	73.5	83.4	85.4	12.3	79.8	88.8	88	12.2	75.6	84.5	83.4	9.3	0.20	0.149	0.212	15	37.2	74.8	71.2	13.8
SVM	70.3	84.1	84.6	13	71.9	88.4	88.8	13.5	77.6	84.3	84.1	7.8	0.19	0.199	0.122	11	41.01	77.5	77.8	6.6
A-DT	79.2	84.3	81.5	12.6	79.8	88.6	84.3	13.7	78.9	82.1	79.2	13.6	0.24	0.171	0.173	15	38.4	63.2	68.4	15.3
A-SVM	70.3	83.6	74.6	17.3	78.8	89.1	80.3	13.4	68.3	80	66.3	18	0.18	0.201	0.211	14.1	38.2	67.9	62.3	15.5
A-NB	65.6	62.3	44.5	19	77.8	43.8	32.6	18.3	19	40.2	45.4	19	0.19	0.203	0.241	16.6	32.5	47.4	39.4	19
A-LR	76.8	85.3	84.5	10	82.1	87.6	88.9	11.1	77.2	83.7	84.2	9.6	0.22	0.221	0.224	17.6	43.1	79.9	74.6	4.5
XGB	83.2	88.1	85.7	3	90.1	91.1	89.7	3.3	82.7	83.9	85.4	5	0.15	0.091	0.123	3.1	44.2	76.8	73.8	7.5
LGB	85.9	84.8	83.5	8.8	92.8	90.9	88.3	5.3	81.1	82.8	82	10.6	0.17	0.103	0.123	7.5	39.8	76.9	71.1	11.6
GB	83.5	87.6	84.3	5.3	91.2	91	89.7	3.1	79.4	83.1	83.3	10.3	0.16	0.233	0.124	10.6	41.3	76.5	74.5	9.3
CTB	86.7	88.4	86.2	1	93.5	91.1	89.9	1.2	85.7	86.7	87.2	1	0.15	0.091	0.091	1.8	45.16	81.2	78.3	1
RF	86.1	85.6	84.6	5.8	91.8	90.3	88.9	4.5	80.2	85.7	84.3	4.1	0.16	0.093	0.124	5.8	41.48	77.9	78.2	5.1
B-SVM	77.2	85.7	85.1	7.8	85.4	88.4	88.2	11.6	73.2	84.6	86.2	7.3	0.19	0.133	0.126	10.6	42.9	77.2	76.3	6
B-NB	72.4	83.6	84.2	14.1	79.5	88.4	88.2	13.6	75.3	84.2	83.8	10.1	0.20	0.141	0.211	13.8	36.8	71.3	71.2	14.5
B-DT	79.2	86.2	82.3	9.1	91.1	89.1	88.1	8.1	78.3	83	80.3	13	0.16	0.105	0.113	6.3	35.9	77.9	72.2	11.5
B-LR	75.6	87.3	83.8	9.3	82.1	88.6	88.6	10	76.2	84.2	84.3	8	0.16	0.093	0.112	4.5	42.3	79.6	73.5	7.3
B-KNN	78.9	85.7	83.5	9.6	89.3	88.4	89	8.5	81	84	82.9	8.6	0.17	0.102	0.145	8.6	36.8	75.8	75.6	11.1

TABLE 15: Ranks in terms of ACC, AUC, and F1-score and the mean ranks of all the classifiers.

	R_ACC	R_AUC	R_F1	R_BS	R_KS	Mean rank
LR	10.1	11.6	11.8	2.1	5.5	8.22
KNN	8.6	11.3	8.3	10	12	10.04
CART	12.6	15.3	14	15.3	12.3	13.9
NB	12.3	12.2	9.3	15	13.8	12.52
SVM	13	13.5	7.8	11	6.6	10.38
A-DT	12.6	13.7	13.6	15	15.3	14.04
A-SVM	17.3	13.4	18	14.1	15.5	15.66
A-NB	19	18.3	19	16.6	19	18.38
A-LR	10	11.1	9.6	17.6	4.5	10.56
XGB	3	3.3	5	3.1	7.5	4.38
LGB	8.8	5.3	10.6	7.5	11.6	8.76
GB	5.3	3.1	10.3	10.6	9.3	7.72
CTB	1	1.2	1	1.8	1	1.2
RF	5.8	4.5	4.1	5.8	5.1	5.06
B-SVM	7.8	11.6	7.3	10.6	6	8.66
B-NB	14.1	13.6	10.1	13.8	14.5	13.22
B-DT	9.1	8.1	13	6.3	11.5	9.6
B-LR	9.3	10	8	4.5	7.3	7.95
B-KNN	9.6	8.5	8.6	8.6	11.1	9.28

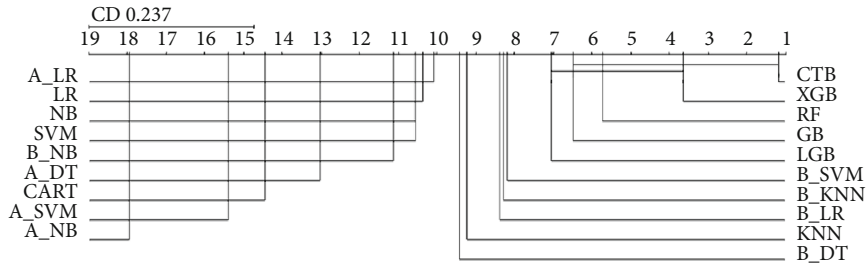


FIGURE 2: Graphical representation of mean ranks using Nemenyi post hoc test.

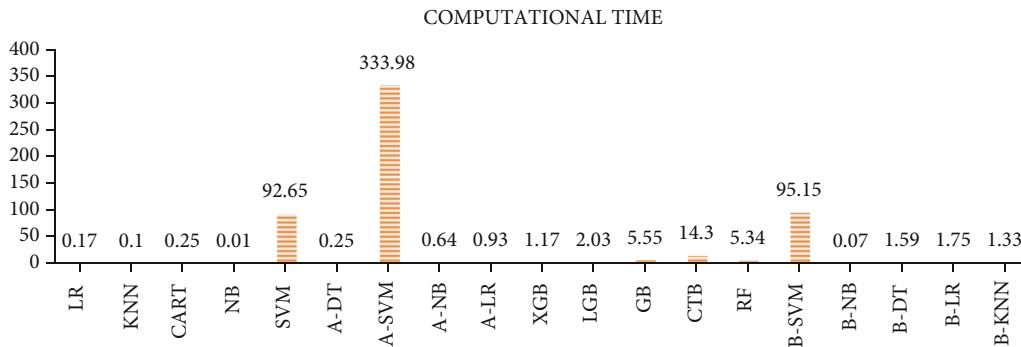


FIGURE 3: Average computational time of base and ensemble models.

proposed to build the credit scoring model. This study develops a hybrid credit scoring model by applying SMOTE and different FS techniques to base and ensemble learners. Three FS techniques are applied in the proposed work, such as IG, GA, and PCA on the balanced training set to select the best predictors. In the experiment, 5 base and 14 ensembles of classifiers are used and the effectiveness of the models is

validated in terms of ACC, AUC, F1-score, BS, and KS metrics across three benchmark credit scoring datasets, i.e., German, Australian, and Japan.

Applying the SMOTE method, we observe some improvements in most of the algorithms, and among them, CTB is the best classifier. Next, FS techniques are applied to all the algorithms and result in further improvements in the performance

of the models. GA is the best FS technique that brings the highest improvements in the model. Therefore, this study suggests that combining the CTB machine learning algorithm with the GA-based FS technique could build an accurate and reliable credit scoring model. The experimental results reveal that all the financial industries could use the proposed hybrid model to predict the defaulters effectively.

The proposed model can be further improved in classification in future studies by incorporating different optimized techniques, such as particle swarm optimization, GA, and ant colony optimization methods. Moreover, multiple base learners can be combined using different ensemble methods, such as random subspace, stacking, and DECORATE. Additionally, more FS techniques, like RELIEF, chi-square, and rough sets, can be applied, and optimized feature subsets can give better results. Finally, more credit scoring datasets should be explored to validate the conclusions of this paper further.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Dongseo University, “Dongseo Cluster Project” Research Fund of 2022 (DSU-20220006).

References

- [1] D. Tripathi, D. R. Edla, and R. Cheruku, “Hybrid credit scoring model using neighborhood rough set and multi-layer ensemble classification,” *Journal of Intelligent Fuzzy Systems*, vol. 34, no. 3, pp. 1543–1549, 2018.
- [2] M. Mihalović, “Performance comparison of multiple discriminant analysis and logit models in bankruptcy prediction,” *Economics & Sociology*, vol. 9, no. 4, pp. 101–118, 2016.
- [3] J. Yi, “Credit scoring model based on the decision tree and the simulated annealing algorithm,” *2009 WRI World Congress on Computer Science and Information Engineering*, vol. 4, no. 2007, pp. 18–22, 2009.
- [4] W. E. Henley and D. J. Hand, “A k-nearest-neighbour classifier for assessing consumer credit risk,” *The Statistician*, vol. 45, no. 1, pp. 77–95, 1996.
- [5] N. C. Hsieh and L. P. Hung, “A data driven ensemble classifier for credit scoring analysis,” *Expert Systems with Applications*, vol. 37, no. 1, pp. 534–545, 2010.
- [6] C. F. Tsai, Y. F. Hsu, and D. C. Yen, “A comparative study of classifier ensembles for bankruptcy prediction,” *Applied Soft Computing*, vol. 24, pp. 977–984, 2014.
- [7] G. Wang, J. Hao, J. Ma, and H. Jiang, “A comparative assessment of ensemble learning for credit scoring,” *Expert Systems with Applications*, vol. 38, no. 1, pp. 223–230, 2011.
- [8] A. Alonso and J. M. Carbo, “Understanding the performance of machine learning models to predict credit default: a novel approach for supervisory evaluation,” *SSRN Electronic Journal*, 2021.
- [9] P. Z. Lappas and A. N. Yannacopoulos, “A machine learning approach combining expert knowledge with genetic algorithms in feature selection for credit risk assessment,” *Applied Soft Computing*, vol. 107, article 107391, 2021.
- [10] C. Oral, “Analytical hierarchy process as a tool for investment appraisal,” *International Journal of Economics and Finance*, vol. 8, no. 4, p. 306, 2016.
- [11] F. A. F. Ferreira, S. P. Santos, and V. M. C. Dias, “An AHP-based approach to credit risk evaluation of mortgage loans,” *International Journal of Strategic Property Management*, vol. 18, no. 1, pp. 38–55, 2014.
- [12] P. Pławiak, M. Abdar, and U. Rajendra Acharya, “Application of new deep genetic cascade ensemble of SVM classifiers to predict the Australian credit scoring,” *Applied Soft Computing*, vol. 84, article 105740, 2019.
- [13] A. Kim and S. B. Cho, “An ensemble semi-supervised learning method for predicting defaults in social lending,” *Engineering Applications of Artificial Intelligence*, vol. 81, pp. 193–199, 2019.
- [14] L. Zhou, D. Lu, and H. Fujita, “The performance of corporate financial distress prediction models with features selection guided by domain knowledge and data mining approaches,” *Knowledge-Based Systems*, vol. 85, pp. 52–61, 2015.
- [15] F. L. Chen and F. C. Li, “Combination of feature selection approaches with SVM in credit scoring,” *Expert Systems with Applications*, vol. 37, no. 7, pp. 4902–4909, 2010.
- [16] S. Oreski and G. Oreski, “Genetic algorithm-based heuristic for feature selection in credit risk assessment,” *Expert Systems with Applications*, vol. 41, no. 4, pp. 2052–2064, 2014.
- [17] S. Dahiya, S. S. Handa, and N. P. Singh, “A feature selection enabled hybrid-bagging algorithm for credit risk evaluation,” *Expert Systems*, vol. 34, no. 6, article e12217, 2017.
- [18] A. Guzmán-Ponce, R. M. Valdovinos, J. S. Sánchez, and J. R. Marcial-Romero, “A new under-sampling method to face class overlap and imbalance,” *Applied Sciences*, vol. 10, no. 15, p. 5164, 2020.
- [19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [20] G. Wang, J. Ma, L. Huang, and K. Xu, “Two credit scoring models based on dual strategy ensemble trees,” *Knowledge-Based Systems*, vol. 26, pp. 61–68, 2012.
- [21] T. Zhang and G. Chi, “A heterogeneous ensemble credit scoring model based on adaptive classifier selection: an application on imbalanced data,” *International Journal of Finance and Economics*, vol. 26, no. 3, pp. 4372–4385, 2021.
- [22] C. F. Tsai, “Combining cluster analysis with classifier ensembles to predict financial distress,” *Information Fusion*, vol. 16, no. 1, pp. 46–58, 2014.
- [23] F. N. Koutanaei, H. Sajedi, and M. Khanbabaee, “A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring,” *Journal of Retailing and Consumer Services*, vol. 27, pp. 11–23, 2015.
- [24] D. Liang, C. F. Tsai, and H. T. Wu, “The effect of feature selection on financial distress prediction,” *Knowledge-Based Systems*, vol. 73, no. 1, pp. 289–297, 2015.

- [25] M. Ala'raj and M. F. Abbod, "A new hybrid ensemble credit scoring model based on classifiers consensus system approach," *Expert Systems with Applications*, vol. 64, pp. 36–55, 2016.
- [26] J. Abellán and J. G. Castellano, "A comparative study on base classifiers in ensemble methods for credit scoring," *Expert Systems with Applications*, vol. 73, pp. 1–10, 2017.
- [27] H. He, W. Zhang, and S. Zhang, "A novel ensemble method for credit scoring: adaption of different imbalance ratios," *Expert Systems with Applications*, vol. 98, pp. 105–117, 2018.
- [28] Y. Xia, C. Liu, B. Da, and F. Xie, *A Novel Heterogeneous Ensemble Credit Scoring Model Based on Bstacking Approach*, vol. 93, Elsevier Ltd, 2018.
- [29] S. Jadhav, H. He, and K. Jenkins, "Information gain directed genetic algorithm wrapper feature selection for credit rating," *Applied Soft Computing*, vol. 69, pp. 541–553, 2018.
- [30] L. Munkhdalai, T. Munkhdalai, O. E. Namsrai, J. Y. Lee, and K. H. Ryu, "An empirical comparison of machine-learning methods on bank client credit assessments," *Sustainability*, vol. 11, no. 3, p. 699, 2019.
- [31] X. Chen, S. Li, X. Xu, F. Meng, and W. Cao, "A novel GSCI-based ensemble approach for credit scoring," *IEEE Access*, vol. 8, pp. 222449–222465, 2020.
- [32] Y. Song, Y. Wang, X. Ye, D. Wang, Y. Yin, and Y. Wang, "Multi-view ensemble learning based on distance-to-model and adaptive clustering for imbalanced credit risk assessment in P2P lending," *Information Sciences*, vol. 525, pp. 182–204, 2020.
- [33] K. Niu, Z. Zhang, Y. Liu, and R. Li, "Resampling ensemble model based on data distribution for imbalanced credit risk evaluation in P2P lending," *Information Sciences*, vol. 536, pp. 120–134, 2020.
- [34] F. Shen, X. Zhao, G. Kou, and F. E. Alsaadi, "A new deep learning ensemble credit risk evaluation model with an improved synthetic minority oversampling technique," *Applied Soft Computing*, vol. 98, p. 106852, 2021.
- [35] R. Emekter, Y. Tu, B. Jirasakuldech, and M. Lu, "Evaluating credit risk and loan performance in online peer-to-peer (P2P) lending," *Applied Economics*, vol. 47, no. 1, pp. 54–70, 2015.
- [36] J. Banasik, J. Crook, and L. Thomas, "Sample selection bias in credit scoring models," *Journal of the Operational Research Society*, vol. 54, no. 8, pp. 822–832, 2003.
- [37] B. Anderson, "Using Bayesian networks to perform reject inference," *Expert Systems with Applications*, vol. 137, pp. 349–356, 2019.
- [38] J. López and S. Maldonado, "Profit-based credit scoring based on robust optimization and feature selection," *Information Sciences*, vol. 500, pp. 190–202, 2019.
- [39] F. Antunes, B. Ribeiro, and F. Pereira, "Probabilistic modeling and visualization for bankruptcy prediction," *Applied Soft Computing*, vol. 60, pp. 831–843, 2017.
- [40] L. Yu, R. Zhou, L. Tang, and R. Chen, "A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data," *Applied Soft Computing*, vol. 69, pp. 192–202, 2018.
- [41] C. L. Huang, M. C. Chen, and C. J. Wang, "Credit scoring with a data mining approach based on support vector machines," *Expert Systems with Applications*, vol. 33, no. 4, pp. 847–856, 2007.
- [42] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: a proposed framework and novel findings," *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 485–496, 2008.
- [43] X. Feng, Z. Xiao, B. Zhong, J. Qiu, and Y. Dong, "Dynamic ensemble classification for credit scoring using soft probability," *Applied Soft Computing*, vol. 65, pp. 139–151, 2018.
- [44] S. Y. Kim and A. Upneja, "Predicting restaurant financial distress using decision tree and AdaBoosted decision tree models," *Economic Modelling*, vol. 36, pp. 354–362, 2014.
- [45] J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: a new classifier ensemble method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1619–1630, 2006.
- [46] A. Chopra and P. Bhilare, "Application of ensemble models in credit scoring models," *Business Perspectives and Research*, vol. 6, no. 2, pp. 129–141, 2018.
- [47] A. Behr and J. Weinblat, "Default patterns in seven EU countries: a random forest approach," *International Journal of the Economics of Business*, vol. 24, no. 2, pp. 181–222, 2017.
- [48] H. Faris, R. Abukhurma, W. Almanaseer et al., "Improving financial bankruptcy prediction in a highly imbalanced class distribution using oversampling and ensemble learning: a case from the Spanish market," *Progress in Artificial Intelligence*, vol. 9, no. 1, pp. 31–53, 2020.
- [49] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [50] M. J. Kearns, R. E. Schapire, and L. M. Sellie, "Toward efficient agnostic learning," *Machine Learning*, vol. 17, no. 2-3, pp. 115–141, 1994.
- [51] P. Pławiak, M. Abdar, J. Pławiak, V. Makarenkov, and U. R. Acharya, "DGHNL: a new deep genetic hierarchical network of learners for prediction of credit scoring," *Information Sciences*, vol. 516, pp. 401–418, 2020.
- [52] N. Arora and P. D. Kaur, "A Bolasso based consistent feature selection enabled random forest classification algorithm: an application to credit risk assessment," *Applied Soft Computing*, vol. 86, article 105936, 2020.
- [53] A. Khashman, "Neural networks for credit risk evaluation: investigation of different neural models and learning schemes," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6233–6239, 2010.
- [54] K. Tran, T. Duong, and Q. Ho, "Credit scoring model: a combination of genetic programming and deep learning," in *2016 Future Technologies Conference (FTC)*, San Francisco, United States, 2016.
- [55] X. Dastile, T. Celik, and M. Potsane, "Statistical and machine learning models in credit scoring: a systematic literature survey," *Applied Soft Computing Journal*, vol. 91, article 106263, 2020.
- [56] M. Herasymovych, K. Märka, and O. Lukason, "Using reinforcement learning to optimize the acceptance threshold of a credit scoring model," *Applied Soft Computing Journal*, vol. 84, 2019.
- [57] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: adaptive synthetic sampling approach for imbalanced learning. In *IEEE International Joint Conference on Neural Networks, 2008*," in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328, Hong Kong, 2008.
- [58] H. Han, W. Y. Wang, and B. H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning," *Lecture Notes in Computer Science*, vol. 3644, no. PART I, pp. 878–887, 2005.

- [59] M. Dianati, I. Song, and M. Treiber, "An introduction to genetic algorithms and evolution strategies," *Sadhana*, vol. 24, no. 4–5, pp. 293–315, 1999.
- [60] J. Huang and C. X. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 299–310, 2005.
- [61] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A - Mathematical Physical and Engineering Sciences*, vol. 374, p. 2016, 2016.
- [62] J. R. Yao and J. R. Chen, "A new hybrid support vector machine ensemble classification model for credit scoring," *Journal of Information Technology Research*, vol. 12, no. 1, pp. 77–88, 2019.
- [63] P. E. Hart, D. G. Stork, and R. O. Duda, *Pattern Classification*, A Wiley-Interscience Publication, 2001.
- [64] D. Zhang, X. Zhou, S. C. H. Leung, and J. Zheng, "Vertical bagging decision trees model for credit scoring," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7838–7843, 2010.
- [65] A. I. Marqués, V. García, and J. S. Sánchez, "Exploring the behaviour of base classifiers in credit scoring ensembles," *Expert Systems with Applications*, vol. 39, no. 11, pp. 10244–10250, 2012.
- [66] A. M. P. Canuto, M. C. C. Abreu, L. de Melo Oliveira, J. C. Xavier, and M. Santos, "Investigating the influence of the choice of the ensemble members in accuracy and diversity of selection-based and fusion-based methods for ensembles," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 472–486, 2007.
- [67] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system," *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, vol. 9, pp. 785–794, 2016.
- [68] G. Ke, Q. Meng, T. Finley et al., "LightGBM: a highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, pp. 3147–3155, 2017.
- [69] E. Al Daoud, "Comparison between XGBoost, LightGBM and CatBoost using a home credit dataset," *International Journal of Computer and Information Engineering*, vol. 13, no. 1, pp. 6–10, 2019.
- [70] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.
- [71] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994.