ISSN: 2029-4913 / eISSN: 2029-4921

Article in press

TECHNOLOGICAL and ECONOMIC DEVELOPMENT of ECONOMY

https://doi.org/10.3846/tede.2025.23060

A HYBRID CLUSTERING AND BOOSTING TREE FEATURE SELECTION (CBTFS) METHOD FOR CREDIT RISK ASSESSMENT WITH HIGH-DIMENSIONALITY

Jianxin ZHU[1,2], Xiong WU[1,2], Lean YU[1,2,3]✉, Xiaoming ZHANG[4]✉✉

[1] *School of Economics and Management, Harbin Engineering University, Harbin, China*
[2] *Key Laboratory of Big Data and Business Intelligence Technology, Harbin Engineering University, Ministry of Industry and Information Technology, Harbin, China*
[3] *Business School, Sichuan University, Chengdu, China*
[4] *School of Information Management and Mathematics, Jiangxi University of Finance and Economics, Nanchang, China*

Article History:
- received 31 October 2023
- accepted 15 November 2024
- first published online 12 February 2025

Abstract. To solve the high-dimensional issue in credit risk assessment, a hybrid clustering and boosting tree feature selection method is proposed. In the hybrid methodology, an improved minimum spanning tree model is first used to remove redundant and irrelevant features. Then three embedded feature selection approaches (i.e., Random Forest, XGBoost, and AdaBoost) are used to further enhance the feature-ranking efficiency and obtain better prediction performance by applying the optimal features. For verification purpose, two real-world credit datasets are used to demonstrate the effectiveness of the proposed hybrid clustering and boosting tree feature selection (CBTFS) methodology. Experimental results demonstrated that the proposed method is superior to others classic feature selection methods. This indicates that the proposed hybrid clustering and boosting tree feature selection method can be used as a promising tool for solving high-dimensional issue in credit risk assessment.

Keywords: feature selection, high-dimensionality, credit risk, minimum spanning tree.

JEL Classification: C44, C53, E51, G21, G32.

✉Corresponding author. E-mail: *yulean@amss.ac.cn*
✉✉Corresponding author. E-mail: *zhxmdy@163.com*

1. Introduction

Credit risk assessment has become a prominent topic for both academic researchers and business practitioners (Yu et al., 2022). The main aim of credit risk evaluation is to predict whether an applicant will default in the future. Misclassifying bad credit as good credit is particularly problematic, as it can lead to significant economic losses for banks and other financial institutions. Therefore, developing an effective credit risk assessment model is essential to minimize potential losses. Over recent decades, various credit classification models have been employed, which is categorized into traditional methods and artificial intelligence (AI) methods.

Traditional methods, such as k-nearest neighbor (*k*-NN), linear discriminant analysis (LDA) (Huang et al., 2022), and decision trees (DT), have been widely used in credit risk assessment by financial institutions. However, these methods rely on certain assumptions about feature variables, which can hinder improvements in model accuracy. Consequently, AI-based models, including support vector machines (SVM) (Maldonado et al., 2017), artificial neural networks (ANN) (Costea et al., 2017), genetic algorithms (GA) (Norat et al., 2023), clustering learning (Baser et al., 2023), and deep learning (Gunnarsson et al., 2021), have been introduced. Compared to traditional methods, AI techniques effectively handle large-scale, nonlinear problems. Beyond individual models, ensemble and hybrid models, which offer higher accuracy than individual models, are utilized in credit risk assessment (Belás et al., 2018; Gonçalves et al., 2016). These models combine the strengths of various classifiers to enhance performance. Typical ensemble learning techniques such as AdaBoost (Sankhwar et al., 2020), XGBoost (Yun et al., 2021), Bagging (Niu et al., 2020), and Random Forest (RF) (Rao et al., 2020) are widely applied in many different areas.

The studies indicated that no single classification model can consistently perform well across all datasets, primarily due to the inherent traits of the data. Real-world credit datasets exhibit traits such as data sparsity, class imbalance, data scarcity, and high dimensionality (Zhang & Yu, 2024). High dimensionality in credit datasets often results in increasing computational complexity and can exacerbate the "curse of dimensionality". The existing solutions are primarily categorized into feature selection (FS) (including filter, wrapper, and embedded methods) and feature extraction (FE). For example, a novel credit risk assessment model is proposed by using a hierarchical attention method to enhance important features, integrate multi-view data, and manage feature acquisition costs for improved performance (Liu et al., 2024). Additionally, an improved multilayer restricted Boltzmann machine (RBM) FE method is proposed to address high-dimensional issue in credit risk assessment, demonstrating significant performance improvements on real-world datasets (Zhu et al., 2024). However, notable limitations are observed in current FS and FE methods for high-dimensional data. These techniques are prone to be overfitting, often fail to effectively eliminate redundant features, and frequently do not capture the relationships between features in complex credit datasets. As a result, the accuracy of the classifier and the quality of decision-making are adversely affected. To overcome these challenges, a novel hybrid clustering and boosted tree feature selection (CBTFS) method has been proposed, with the aim of improving credit risk assessment by efficiently addressing high-dimensionality issue, thereby enhancing prediction accuracy.

To address the challenges posed by high-dimensional data in credit risk assessment, a novel hybrid CBTFS method is introduced. This approach begins with an improved minimum spanning tree model, which efficiently eliminates redundant and irrelevant features. Subsequently, three embedded feature selection algorithms – RF, XGBoost, and AdaBoost – are employed to identify the highest-ranked features from the aforementioned methods. This integration aims to formulate an optimal feature set, effectively achieving the goal of "selecting the best among the best" and enhancing prediction performance. Furthermore, the proposed hybrid CBTFS method is experimentally verified to be an effective solution for addressing high-dimensional challenges in credit risk assessment in this paper.

The main contributions of this paper are summarized into two-fold. On the one hand, a new framework for credit risk assessment is first proposed, integrating a clustering technique (i.e., an improved minimum spanning tree (IMST)), and three hybrid feature selection methods based on boosted tree modeling. This framework addresses feature redundancy in high-dimensional data through a hybrid CBTFS method, thereby enhancing prediction performance. On the other hand, the framework employs improved MST along with several classical clustering methods and combines them with three embedded feature selection methods – RF, XGBoost, and AdaBoost – to increase feature ranking efficiency and eliminate a significant number of redundant features. This hybrid clustering model does not only remove redundant and irrelevant features but also aids in setting effective thresholds for MST, thus improving the model's clustering performance. Thus, the proposed hybrid method effectively addresses credit risk assessment challenges associated with high dimensionality.

The primary motivation of this paper is to propose a hybrid CBTFS method for credit risk assessment with high dimensionality, and attempt to improve the classification predictive performance of high-dimensional sample modeling. The rest of the paper is structured as follows. The literature review is described Section 2. Section 3 recommends the components of the proposed hybrid CBTFS method in detail. Section 4 presents the experimental study. Section 5 presents the experimental results by describing performance evaluation and comparative analysis. Section 6 concludes the paper and meantime provides guidelines for future work.

2. Literature review

In the big data era, financial institutions increasingly contend with high-dimensional datasets due to the vast array of data attributes available from credit applicants. However, these datasets often contain redundant and irrelevant features, which can diminish the accuracy of classifiers and increase computational complexity. To mitigate these issues, FE (Tsafrir et al., 2023) and FS (Sun et al., 2015; Zhang et al., 2022) methods are commonly employed.

FE transforms the original data into a new space by using mathematical techniques, improving training efficiency and predictive accuracy. For example, PCA has been used for credit risk classification (Yu et al., 2021), but it can be ineffective when the feature dimensions exceed the sample size, leading to a loss of interpretability and data integrity.

Generally speaking, FS aims to retain essential information while reducing dimensionality and is generally categorized into filters, wrappers, and embedded methods. Some advanced techniques such as feature clustering and minimum spanning tree (MST) (Liu et al., 2022) offer additional dimensionality reduction capabilities. Feature clustering identifies redundant features and enhances model interpretability by grouping similar features, using methods like Birch, Spectral Clustering (SC) (Yang et al., 2023), K-modes, K-means, K-means++ (Li & Wang, 2023), Agglomerative Clustering (AC) (Jáñez-Martino et al., 2023), and K-prototypes (Kuo & Wang, 2022). However, these methods face challenges related to similarity measures, scalability, and cluster interpretation. MST facilitates dimensionality reduction by identifying a minimal subset of edges in a weighted graph that connects all vertices with minimal total edge weight. Although algorithms like Kruskal and Prim are computationally efficient, MST's emphasis on edge weight minimization may neglect factors such as reliability and latency, and its performance can be sensitive to the accuracy of edge weights. Balancing the strengths and limitations of these FS methods is crucial for effective feature selection. Therefore, to enhance organization and provide a concise overview of prior research, Table 1 was introduced to compare the primary attributes of previous studies, emphasizing their definitions. These methodologies are outlined in Table 1, with each category highlighting its unique strengths.

Table 1. Typical feature selection methods

Methods	Definitions	Researches
Filters	Feature importance is assessed using statistical modeling to identify the most relevant features.	Li et al. (2024b); Macedo et al. (2022); Maldonado et al. (2017); Ouaderhman et al. (2024); Sankhwar et al. (2020)
Wrappers	Feature are assessed in conjunction with classifier performance.	Chandrashekar and Sahin (2014); Chaudhuri (2024); Li et al. (2024a); Zhao et al. (2023); Zorarpaci (2024)
Embedded methods	The most relevant features are selected during model training.	Hu et al. (2023); Kozodoi et al. (2019); Qian et al. (2022); Tsai et al. (2021); Zhao et al. (2023)
Ensemble methods	Integrating feature selection results from multiple methods.	He et al. (2018); Osanaiye et al. (2016); Seijo-Pardo et al. (2017); Song et al. (2013); Tsai et al. (2024); Zhang et al. (2015)
Hybrid methods	The methods are combined to enhance accuracy and efficiency by statistical measures and model-based techniques.	Wang et al. (2018b); Naseriparsa et al. (2013); Pashaei and Pashaei (2022); Sahu and Dash (2024)

In summary, FE and FS, including feature clustering and MST, are critical for managing high-dimensional data. These methods effectively reduce dimensionality, eliminate redundancy, and enhance model performance, although they encounter challenges related to interpretability, scalability, and accuracy. Existing FS and feature clustering methods in credit risk assessment reveal several significant issues. First, these techniques are prone to be overfitting and can be time-consuming, necessitating rigorous validation, particularly in high-dimensional contexts – an area often overlooked in previous research. Second, current methods often fail to eliminate redundant and irrelevant features in high-dimensional datasets, which adversely affects classifier accuracy. Financial institutions frequently encounter credit datasets characterized by sparsity and class imbalance, where irrelevant features introduce noise and obscure meaningful patterns. Traditional FS methods struggle to capture the complex relationships among features, underscoring the need for advanced methods that effectively identify essential features while discarding those that do not contribute to predictive power. Finally, many feature clustering methods encounter challenges related to scalability and clarity in cluster formation, complicating the analysis of complex credit datasets. Although techniques such as K-means and MST are capable of grouping similar features, their significance is often not conveyed clearly, which hinders informed decision-making in credit risk assessment. Therefore, the development of clustering methods that effectively scale and provide clear insights is essential for improving these outcomes. To address these challenges, this paper proposes a hybrid CBTFS method. This methodology includes a preprocessing phase, utilizes multiple feature clustering methods along with the IMST for initial FS, and employs powerful boosting trees for further refinement. The effectiveness of this method is assessed by using high-dimensional credit datasets and relevant evaluation criteria.

3. The proposed hybrid CBTFS method

In this section, a hybrid CBTFS paradigm is proposed to address the high dimensionality challenge in credit classification. The general framework of the proposed method is illustrated in Figure 1.

As can be seen from Table 1, this hybrid method incorporates the IMST model, which is designed to reduce computational time within a hybrid CBTFS structure. At the same time, three advanced boosting tree algorithms – RF, XGBoost, and AdaBoost – are employed due to their robust learning capabilities and their use of feature importance techniques to select valuable feature information effectively. This combination is intended to enhance processing efficiency and improve the quality of feature selection, ultimately leading to superior classification performance.

As shown in Figure 1, four main stages, preprocessing and partitioning data, clustering based feature selection, boosting tree-based feature selection, and final output results are included. To clearly articulate the details of the hybrid CBTES method, the operational steps of the proposed hybrid CBTFS can be presented by Algorithm 1. Furthermore, detailed description of the four stages and models used are given in the following Sections 3.1–3.4, respectively.

Algorithm 1 The proposed hybrid CBTFS method

```
Input: high dimensional credit datasets
Output: optimal feature subset
Procedure hybrid CBTFS
    Stage 1: Preprocessing and partitioning data
        for data preprocessing do
        The mean imputation and standardization of data
        Divide into training set and testing set
        end for
    Stage 2: Clustering based feature selection
        for multiple clustering models do
        Choosing multiple clustering models for ensemble
        Generating optimal clusters using the DBI model
        Generating clustering results $\pi^*(D)$ using the consistency function
        end for
        for improved minimum spanning tree clustering model do
        Constructing the Minimum Spanning Tree (MST) Network
        Disconnect the node and set the threshold $\tau$
        Obtaining multiple feature subsets
        end for
        end for
    Stage 3: Boosting tree-based feature selection
        Three boosting trees were employed to evaluate feature importance
        Obtaining optimal subset of features for each case.
        end for
    Stage 4: Final output results
        The results from the Stage 3 form the new optimal subset
        Classification by classifiers
        end for
end procedure
```

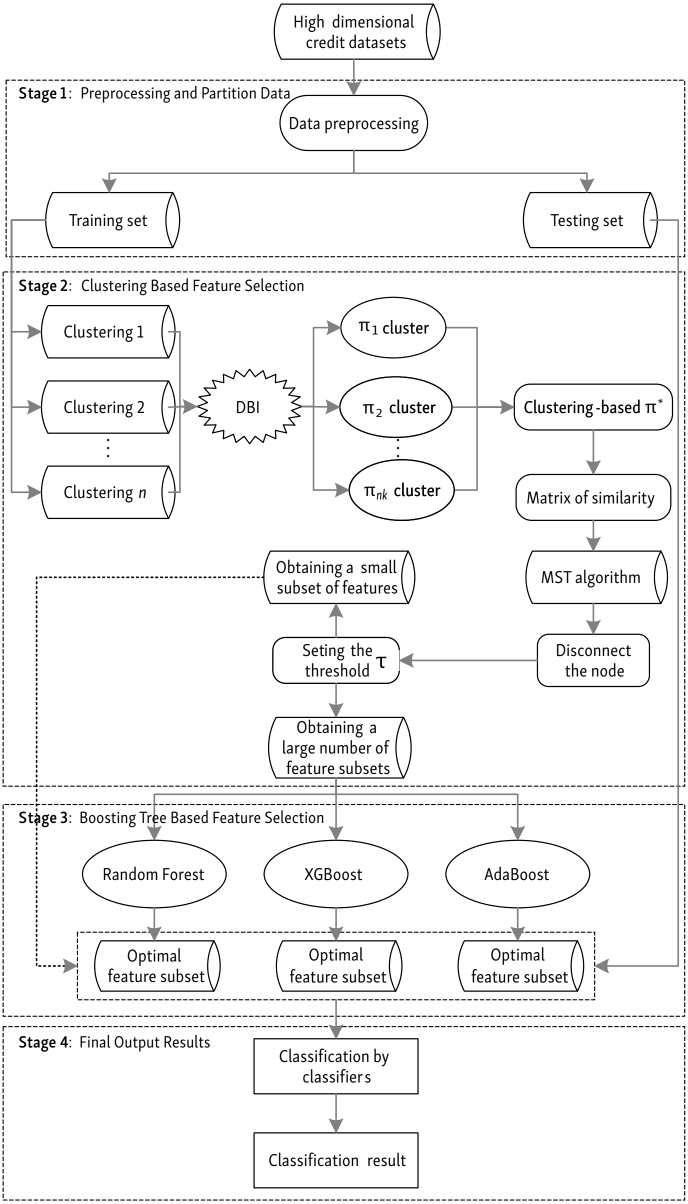

Figure 1. General framework for a hybrid CBTFS method

3.1. Preprocessing and partitioning data

In this Section, the dataset was split into training and testing sets in an 80/20 ratio (He et al., 2018). First, the mean is employed to impute missing values in the dataset. Second, the credit dataset is processed by using normalization techniques, where the credit dataset is mapped to a range between 0 and 1, as shown in Equation (1) below.

$$x_i^* = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}, \tag{1}$$

where x_i and x_i^* represent the values before and after the normalization of the sample data, respectively. $x_{\min}$ and $x_{\max}$ denote the minimum and maximum values in the sample data, respectively.

Finally, the Synthetic Minority Oversampling Technique (SMOTE) and Edited Nearest Neighbor (ENN) methods were used to balance the two classes (Xie et al., 2019). Experiments were conducted ten times to obtain average prediction results.

3.2. Clustering based feature selection

This section outlines how combining multiple clustering models with IMST leverages their strengths to select the most informative and least redundant features. Feature clustering, unlike conventional methods, reduces dimensionality to enhance model efficiency and speed while mitigating overfitting. It groups related features to aid in feature engineering, creating more relevant and informative features, and improves model performance by eliminating redundancy and highlighting key patterns. This method is essential for achieving high classification accuracy in complex, high-dimensional datasets, as detailed in Subsections 3.2.1–3.2.2.

3.2.1. Multiple clustering models for feature selection

To mitigate the bias introduced by using a single clustering algorithm in feature selection, a technique integrating multiple clustering algorithms is proposed. This approach consists of three primary steps.

Step 1: Selecting multiple clustering models for the ensemble. In this paper, seven clustering models (SC, K-means++, AC, Birch, K-prototypes, K-means, and K-modes) are applied to the training datasets to obtain the optimal clustering subsets, denoted as π_{n-k}, where n is the number of models and k is the number of clusters.

Step 2: Optimal clusters are generated by using the Davies-Bouldin Index (DBI) model (Ros et al., 2023). The DBI, which measures the average maximum similarity within clusters, is used to formulate multiple optimal clusters. These clusters are combined into a base clustering, denoted as π^*. The value of k in π_{n-k} is determined by the DBI model. For n-dimensional points, let C_i represent a cluster, and X_j denote an n-dimensional feature vector assigned to cluster C_i.

$$S_i = \left(\frac{1}{T_i} \sum_{j=1}^{T_i} X_j - A_{ip}^{q} \right)^{1/q}, \tag{2}$$

where A_i represents the centroid of cluster C_i, and T_i denotes its size. S_i is the q^{th} root of the q^{th} moment of the points in cluster C_i about the mean. Typically, p is set to 2, making this distance a Euclidean metric. It is crucial that the distance metric used aligns with that of the clustering algorithm to ensure meaningful results.

$$M_{i,j} = \left\| A_i - A_j \right\|_p \left(\sum_{k=1}^{N} \left| a_{k,i} - a_{k,j} \right|^p \right)^{1/p}, \tag{3}$$

where $M_{i,j}$ quantifies the separation between clusters C_i and C_j. In the n-dimensional centroid A_i, $a_{k,i}$ denotes the k^{th} element, where k indexes the data features, with n elements in total. The effectiveness of the clustering scheme is evaluated by R_{ij}, which maximizes $M_{i,j}$ and minimizes S_i, the scatter within cluster C_i. The DBI is then calculated as the ratio of S_i to $M_{i,j}$, as shown in the following Equation.

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}, \tag{4}$$

where S_i or S_j denotes the diameter of the class i or j, M_{ij} denotes the distance between the centroid of class i and j. Through the above formula, the maximum value $R_i = \max(R_{ij})$ is selected from $R_{ij}\left(i \neq j\right)$, the value of the largest similarity in the similarity between class i and other classes. Finally, the mean of these maximum similarities for each class is calculated to obtain the DBI value.

$$\mathrm{DBI} = \frac{1}{N}\sum_{i=1}^{N} R_i, \tag{5}$$

where N represents the number of classes. The smaller the DBI value, the better the clustering results.

Step 3: Clustering results $\pi^*(D)$ are generated by using a consistency function. Co-matrix-based methods are currently the most effective consistency functions. The DBI value guides the selection of optimal π_{n-k} clusters. Let D represent a dataset, the i^{th} clustering of D, π_i (D), is defined by:

$$\pi_i^*\left(D\right) = \left\{C_{i1}, C_{i2}, \dots, C_{i\left|\pi_i^*(D)\right|}\right\}, \tag{6}$$

where C_{ij} denotes the j^{th} cluster of $\pi_i^*\left(D\right)$, where $1 \leq j \leq \left|\pi_i^*\left(D\right)\right|$, and satisfies $D = \bigcup_{j=1}^{\left\|\pi_i(D)\right\|} C_{ij}$. For example, given a set of clustering, $\prod\left(D\right) = \left\{\pi_1\left(D\right), \pi_2\left(D\right), \dots, \pi_n\left(D\right)\right\}$, it is essential to identify the optimal cluster $\pi^*\left(D\right) = \left\{C_1^*, C_2^*, \dots, C^*_{\left|\pi_i^*(D)\right|}\right\}$. Therefore, an $n \times n$ co-matrix CO is constructed, where each element of the matrix is represented by:

$$CO\left(x_i, x_j\right) = 1/M\sum_{m=1}^{M} S_m\left(x_i, x_j\right)\left(1 = i, j \leq n\right), CO\left(x_i, x_j\right) \in \left[0,1\right], \tag{7}$$

where M represents the total number of clusters and m denotes a specific cluster within the clustering. $CO\left(x_i, x_j\right)$ represents the similarity between samples x_i and x_j, resulting in a symmetric similarity matrix. In this matrix, $v_m\left(x_i\right)$ denotes the class label of sample x_i in the base clustering π_i, if $v_m\left(x_i\right) = v_m\left(x_j\right)$, then $S_m\left(x_i, x_j\right) = 1$, otherwise $S_m\left(x_i, x_j\right) = 0$. Since CO is a co-matrix, it can serve as input for any similarity-based clustering model to generate the final clustering result $\pi^*(D)$.

In summary, the three steps outlined above effectively cluster features, it does not only enhance dimensionality reduction, interpretability, and model performance, but also deepens data understanding, making it a valuable technique for improving data quality and optimizing machine learning results.

3.2.2. Improved minimum spanning tree clustering model for feature selection

The minimum spanning tree (MST) is a fundamental concept in graph theory, widely recognized for its ability to represent relationships within sets of data. The MST algorithm minimizes the total edge weights, a principle analogous to clustering, which groups data points based on their similarity. This makes MST effective in capturing data relationships and aiding in the process of clustering. However, MST clustering faces some limitations. On the one hand, it is sensitive to outliers and noise. On the other hand, it struggles with high-dimensional datasets due to the "curse of dimensionality". To address these limitations, the improved MST (IMST) algorithm was developed. IMST leverages the similarity matrix generated by the feature clustering model to optimize feature subsets and effectively manage the complexities of high-dimensional data clustering. Algorithm 2 is the pseudo-code for the IMST model.

Algorithm 2: Improved Minimum spanning tree clustering model

```
Input: Weighted connected graph $G = (V, E)$, where $V$ is the vertex set and $E$ is the edge set.
Output: Some small subset of features and a large number of feature subsets;
1. Initialization: $V_{new} = \{X\}$, X is any node in set $V$, $E_{new} = V$;
2. Repeat steps 2-4 until $V = E_{new}$;
3. Select edge $(u, v)$ with the minimum weight in set $E$, where $u \in V_{new}$ and $V$ is not included in $V_{new}$ (if
   there are multiple edges that meet the above conditions, then select the edge randomly);
4. Add $v$ to $V_{new}$ and $(u, v)$ to $E_{new}$;
5. Obtaining updated $V_{new}$;
6. Nodes X are disconnected using pruning technique, resulting in the automatic generation of multiple
   clusters;
7. Setting threshold $\tau$ for dividing into clusters (a small subset of features and a large number of feature
   subsets).
```

To enhance the clarity and effectiveness of the improved IMST methodology, three key improvements are proposed in this paper.

First, pruning technique is introduced to disconnect tree nodes that link clusters, effectively reducing the complexity and improving the performance of the IMST in high-dimensional settings. This pruning process ensures that the IMST can identify and separate clusters more accurately, even when the data is dense and features are numerous. By isolating clusters that are connected through weak links, a clearer and more meaningful clustering structure can be obtained.

Second, the IMST method is designed to identify both clusters with many features and those with few features. This dual capability is essential for high-dimensional data where feature distribution can vary widely. Clusters with a large number of features are treated differently than those with fewer features, ensuring that the FS process is balanced and comprehensive.

Finally, to address MST's sensitivity to outliers, a threshold parameter τ is introduced to distinguish between majority and minority clusters. Minority clusters, identified as outliers, are directly included in the optimal feature subset for the boosting tree-based FS method. This inclusion ensures valuable but sparse features are retained. Majority clusters are processed by using three robust boosting tree models: RF, XGBoost, and AdaBoost, known for their effectiveness in high-dimensional data. Integrating these models can enhance the accuracy and robustness of feature selection.

The IMST method effectively addresses the challenges of high-dimensional data and outliers by employing pruning techniques and a threshold parameter τ. By distinguishing between majority and minority clusters, IMST ensures the retention of crucial features, thereby enhancing the quality of feature selection. The integration of robust boosting tree models, such as RF, XGBoost, and AdaBoost, further refines feature subsets, leading to improved model accuracy and robustness. These enhancements enable IMST as a powerful tool for FS in complex, high-dimensional datasets.

3.3. Boosting tree-based feature selection

Feature selection (FS) is a critical step of machine learning, and three prominent tree-based ensemble models – Random Forest (RF), Extreme Gradient Boosting (XGBoost), and Adaptive Boosting (AdaBoost) – are particularly effective for this purpose. Each model possesses unique strengths that enhance the feature selection process. By integrating these models, their combined strengths are leveraged for a robust FS process. Feature importance scores are combined in this ensemble method, yielding a more reliable and informative subset of features. The balanced feature importance of these methods enhances model accuracy and generalizability, making it proficient in handling complex, high-dimensional datasets.

Furthermore, the advantages of RF, XGBoost, and AdaBoost are considered complementary, rendering them particularly effective for feature selection. Overfitting is reduced and feature importance is evaluated through the ensemble method of RF, ensuring that only the most relevant features are retained. Accuracy is improved by XGBoost through the sequential correction of errors and the incorporation of regularization techniques, which help to maintain model performance. The capacity of AdaBoost to focus on misclassified instances enables significant features to be highlighted, which might otherwise be overlooked. By leveraging the strengths of these three methods, a more reliable and informative subset of features can be obtained, ultimately enhancing model accuracy and generalizability.

Connecting the FS process with the results from the previous stage is essential for ensuring a systematic approach. In this integration, the majority feature clusters are input into RF, XGBoost, and AdaBoost, enabling these models to select significant features based on their importance rankings. This connection does not only facilitate a refined selection process but also prioritizes relevant features while maintaining model performance. By systematically incorporating insights from the previous stage, the coherence and effectiveness of the FS process can be ensured.

The significance of RF, XGBoost, and AdaBoost lies in their capacity to effectively handle complex, high-dimensional datasets. The rationale for selecting the top 10%, 20%, and 30% features in each majority cluster is considered to be multi-faceted. First, these thresholds facilitate a systematic examination of feature importance, prioritizing the most relevant features while managing model complexity, which is essential for high predictive performance. Second, using multiple percentage thresholds accounts for variability in feature significance across clusters, thereby improving model generalization. Selecting the top 10% focuses on critical features, reducing computational complexity, while including the top 20% and 30% allows for broader exploration without compromising key features. Finally, integrating top-ranked features with minority clusters ensures a comprehensive feature set, combining impactful and unique features and leveraging the strengths of boosting tree models. This stepwise approach accommodates varying degrees of feature importance, leading to more accurate and reliable classifications, especially in applications like credit risk assessment.

3.4. Final output results

After obtaining the optimal input feature set from the CBTFS method, a classification strategy needs to be selected from a variety of individual classifiers and ensemble classifiers for final output results. Three individual classifiers, decision tree (DT), *k*-nearest neighbor (KNN), and naïve Bayes (NB) (Niu et al., 2020) are chosen as the single classifier model for the optimal feature subset. In the meantime, to enhance comparison and achieve balanced results, ensemble models such as RF, AdaBoost, and XGBoost (Avuçlu, 2021) are also employed. Combining these ensemble methods with individual classifiers generally improves classification performance under similar conditions.

4. Experimental design

To validate the proposed methodology, two real-world credit datasets were utilized. For comparison purposes, several FS methods were applied, as presented in Table 2.

Table 2. Comparison of FS method categories

Types	FS methods	Researches
Filters	Variance (VAR)	Wang and Hong (2015)
	Relief	Palma-Mendoza et al. (2018)
	ReliefF	Palma-Mendoza et al. (2018)
	Minimum Redundancy Maximum Relevance (mRMR)	Pashaei and Pashaei (2022)
Wrappers	Genetic algorithm (GA)	Norat et al. (2023)
	Whale optimization algorithm (WOA)	Mirjalili and Lewis (2016); Said et al. (2023)
	Particle swarm optimization (PSO)	Tran et al. (2019); Wang et al. (2018a)

Furthermore, Section 4.1 describes the data, while Section 4.2 presents the evaluation criteria.

4.1. Data descriptions

In this experiment, two public high-dimensional credit data sets, China Union Pay (CUP for short) and Kaggle datasets are applied. The CUP credit dataset is obtained from the data competition created by China Union Pay (Zhang et al., 2022), and Kaggle credit risk dataset is got from the UCI Machine Learning Repository (https://www.kaggle.com/jacklizhi/creditcard). The description of the two real-world datasets is listed in Table 3.

Table 3. Description of two real-world high-dimensional credit datasets

Dataset	No. Instances	No. Paid as Agreed	No. Default	No. Total feature
CUP	11,017	8,873	2,144	199
Kaggle	105,471	95,688	9,783	534

Table 3 indicates that the CUP credit dataset comprises 11,017 credit applicants with 199 variables, which is divided into 8,873 'paid as agreed' (80%) and 2,144 'default' (20%) cases. The Kaggle dataset includes 105,471 applicants with 534 features, with 95,688 (90%) classified as 'paid as agreed' and 9,783 (10%) as 'default'. To address class imbalance and ensure comparability, 5,500 samples from each class were randomly selected, aligning with the CUP dataset size for subsequent classification.

4.2. Evaluation metrics

To assess the performance of proposed CBTFS method, accuracy (ACC for short), area under ROC curve (AUC), Precision, and G-means are used as evaluation metrics for credit classifications (Chowdhury et al., 2022). The most commonly applied measure of classification performance is accuracy, which is the percentage of correct predictions. The AUC value of receiver operating characteristic (ROC) ranges from 0.5 to 1, and values above 0.8 can be considered as a good partition between the two classes of the target variable (Görüş et al., 2024). Precision measures the proportion of positively predicted labels that are truly correct, and recall represents the ability to correctly predict the positives out of actual positives. G-means is a harmonic combination of Recall and Precision.

All experimental analyses were conducted on a laptop with an Intel Core i7-9700F 3.00 GHz processor and 16 GB of RAM. The model parameters were set as follows. The KNN's parameter *k* was set to 10, the ensemble learning classifier used 100 trees, and AdaBoost was configured with 100 iterations. Each experiment was repeated 10 times, with results averaged to ensure robustness.

To assess the statistical significance of different FS methods, two tests were performed. A paired *t*-test evaluated the significance of CBTFS across various classification methods, while the non-parametric Wilcoxon test compared different classification models on various datasets.

5. Experimental results

In this Section, experimental results are presented to verify the superiority of the proposed CBTFS method. To highlight the effectiveness of the proposed method in this paper, the corresponding experimental results are presented in Sections 5.1–5.2.

5.1. Results of optimal clustering models selected by Davies Bouldin Index

In this Section, the DBI was utilized over the Elbow method (Liu & Deng, 2021) and Silhouette Analysis (Gramegna & Giudici, 2021) due to the complexities of credit risk data. The DBI measures the ratio of within-cluster scatter to between-cluster separation, focusing on both compactness and separation. The smaller this ratio, the better, as larger distances between classes and smaller distances within classes indicate improved clustering. This measurement is crucial for high-dimensional credit risk data, which often has overlapping and intricate cluster structures. A more nuanced evaluation of clustering performance is provided by the DBI compared to the Elbow method, which mainly focuses on variance explained.

The results of DBI are shown in Figures 2 and 3, where the seven popular clustering models (SC, K-means++, AC, Birch, K-prototypes, K-means, and K-modes) are demonstrated on the coordinates of the horizontal axis and the DBI values are revealed on the vertical axis.

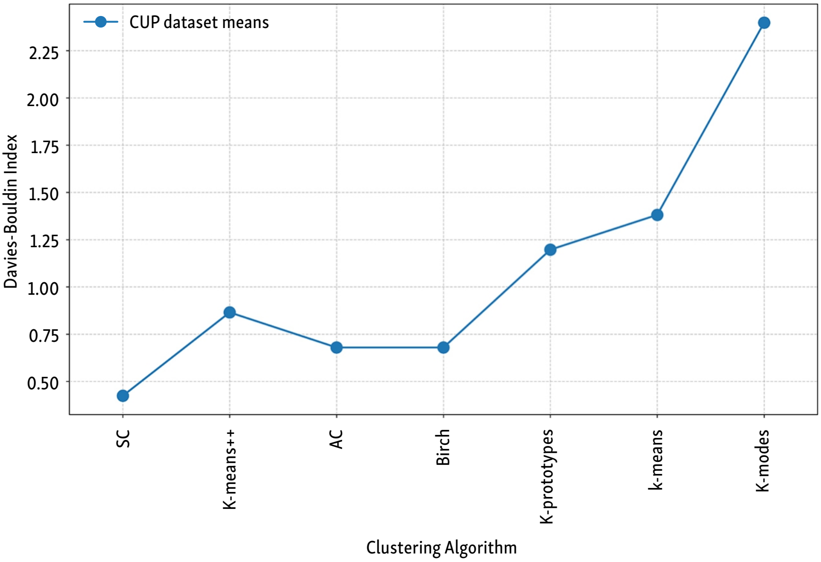

Figure 2. DBI values of the CUP dataset

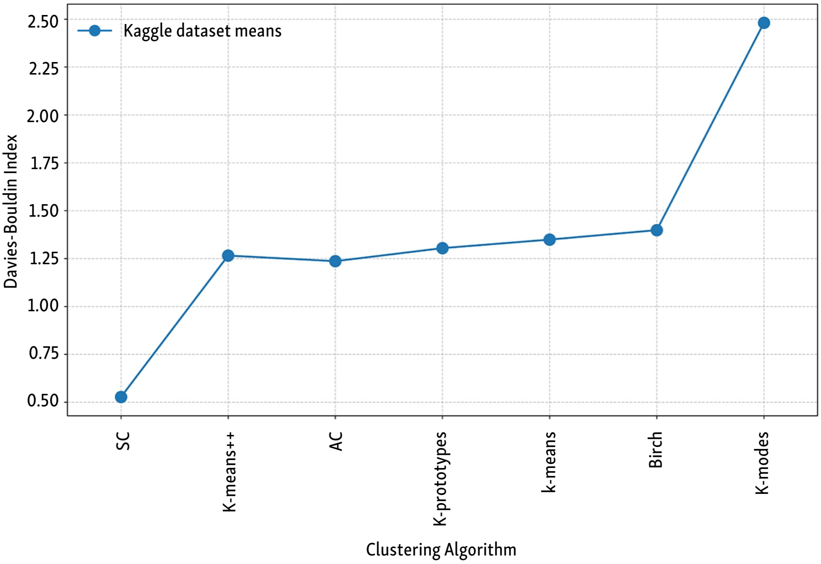

Figure 3. DBI values of the Kaggle dataset

The average values of 10 times in seven popular clustering models are shown in the line chart. As shown in Figures 2, all the DBI values for the top four clustering models (SC, K-means++, AC, and Birch) on the CUP dataset are less than 1, indicating superior clustering performance. The selection of these four models for ensemble clustering is based on the following three critical factors.

First, low DBI values indicate that the models effectively balance within-cluster compactness and between-cluster separation, which is essential for distinguishing credit risk profiles. Second, the diversity among SC, K-means++, AC, and Birch enhances the ensemble model by leveraging various clustering methods. That is, SC captures complex shapes, K-means++ optimizes initial centroids, AC is effective with hierarchical data, and Birch handles large datasets efficiently. Finally, the inclusion of K-prototypes accommodates mixed data types prevalent in credit risk datasets, further validating the model's effectiveness. This integration of methods capitalizes on the strengths of each algorithm, resulting in a robust model for accurate credit risk classification.

Based on the average results of seven clustering models with tested ten times, the top four methods were selected for ensemble clustering (π^*). IMST was then employed to automatically cluster features and identify the most useful ones. In the CUP dataset, 199 features were grouped into 10 clusters with sizes of 1, 5, 1, 25, 116, 2, 1, 25, 4, and 19. With a threshold τ set to 10, six clusters with fewer than 10 features were chosen for their superior discrimination and independence. Consequently, 14 features were selected for further analysis. The remaining clusters (*i.e.*, 25, 116, 25, and 19) proceeded to Stage 3 of the boosting tree-based feature selection, where AdaBoost, RF, and XGBoost were used to identify the top 10 %–30% of features. Finally, features from both steps were combined for classification evaluation.

Similarly, in the Kaggle dataset, 769 features are automatically clustered into 21 clusters (i.e., 28, 21, 72, 3, 3, 65, 54, 85, 27, 257, 8, 4, 11, 110, 2, 4, 2, 5, 5, 2, and 1). A threshold value of 10 effectively separates these clusters. Features in clusters with less than 10 features are first selected, as they exhibit better discrimination and independence. This results in 39 features being selected. The remaining clusters with more than 10 features (i.e., 28, 21, 72, 65, 54, 85, 27, 257, 11, and 110) undergo boosting tree-based feature selection using AdaBoost, RF, and XGBoost to obtain the top 10%–30% features. Finally, the features from both steps are combined for classification evaluation.

5.2. Experimental results of two real-world credit datasets

The experimental results underscore the importance of selecting an optimal number of features. While too few features can reduce model effectiveness, an excessive number can lead to increased computational time. The final feature set, optimized for comparison with other FS methods, strikes a balance between these considerations. The detailed methodology and results are presented in Sections 5.2.1–5.2.3.

5.2.1. Results of the tree-based feature importance ranking method

Each of the four assessment metrics (ACC, Precision, G-mean, and AUC) was evaluated by repeating the experiments 10 times to validate the proposed hybrid CBTFS method. The average results from these repetitions were compared with those of other FS methods. Tables

4–6 present the top 10%, 20%, and 30% features for two real-world credit datasets using the hybrid CBTFS method. The tables are organized by testing set, with the top results highlighted in bold. The first column of each table lists the FS method used in Stage 3, while the second column shows the five individual and four ensemble classification methods applied.

As can be seen from Table 4, three interesting results can be found.

First of all, from the viewpoint of different feature ranking methods, all feature ranking methods enhance prediction performance to some extents. Notably, FS methods effectively address the challenges of high dimensionality. In this experiment, using XGBoost as a ranking method yields the highest performance in ACC, Precision, G-mean, and AUC. This is likely attributable to its gradient boosting technique, which iteratively corrects errors, manages complex feature interactions, and incorporates built-in regularization. Similarly, when AdaBoost is employed, it also performs exceptionally well in ACC, Precision, G-mean, and AUC. This can be attributed to two factors: the top 10% of features selected by XGBoost, and the robust classification capabilities of the XGBoost classifier.

Table 4. The top 10% features selected by the tree-based feature importance ranking methods

FS Methods	Classification Methods	The CUP dataset				The Kaggle dataset			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
AdaBoost	LDA	0.6686	0.9155	0.6989	0.7012	0.4359	0.9099	0.5359	0.5979
	LogR	0.6054	0.9141	0.6634	0.6727	0.2805	0.9230	0.3335	0.5379
	KNN	0.6327	0.9079	0.6723	0.6764	0.4347	0.8177	0.4968	0.5138
	NB	0.6196	0.9153	0.6689	0.6793	0.4510	0.8774	0.5392	0.5776
	DT	0.6544	0.8758	0.6389	0.6414	0.5833	0.8367	0.5535	0.5560
	RF	0.7171	0.9159	0.7223	0.7225	0.6005	0.8559	0.5890	0.5901
	XGBoost	0.7294	0.9116	0.7210	0.7212	0.6730	0.8483	0.5760	0.5920
	AdaBoost	0.6839	**0.9253**	0.7175	0.7204	0.5911	0.8534	0.5837	0.5845
	Bagging	0.7207	0.8997	0.6983	0.6994	0.6341	0.8362	0.5495	0.5623
RF	LDA	0.6392	0.9166	0.6846	0.6900	0.4277	0.9113	0.5250	0.5958
	LogR	0.5999	0.9175	0.6629	0.6741	0.2952	0.9262	0.3480	0.5435
	KNN	0.6079	0.9051	0.6567	0.6631	0.3996	0.8079	0.4727	0.5069
	NB	0.6644	0.9074	0.6819	0.6880	0.4489	0.8762	0.5364	0.5784
	DT	0.6901	0.8828	0.6592	0.6612	0.5755	0.8234	0.5380	0.5413
	RF	0.7155	0.9173	0.7235	0.7238	0.5812	0.8538	0.5886	0.5894
	XGBoost	0.7350	0.9046	0.7115	0.7126	0.6642	0.8459	0.5819	0.5945
	AdaBoost	0.7115	0.9200	**0.7253**	**0.7259**	0.5783	0.8534	0.5870	0.5882
	Bagging	0.7152	0.8972	0.6922	0.6933	0.6386	0.8289	0.5417	0.5586
XGBoost	LDA	0.6686	0.9136	0.6966	0.6987	0.4093	0.9167	0.5110	0.5914
	LogR	0.6229	0.9122	0.6713	0.6779	0.3322	**0.9420**	0.4174	0.5680
	KNN	0.6066	0.8980	0.6490	0.6539	0.4036	0.8201	0.4815	0.5150
	NB	0.6819	0.9047	0.6913	0.6918	0.4989	0.8747	0.5703	0.5892
	DT	0.7093	0.8825	0.6632	0.6671	0.5782	0.8356	0.5525	0.5541
	RF	0.7245	0.9075	0.7126	0.7129	0.5692	0.8575	0.5854	0.5862
	XGBoost	**0.7452**	0.8997	0.7059	0.7086	**0.6803**	0.8557	**0.5969**	**0.6085**
	AdaBoost	0.7188	0.9126	0.7182	0.7183	0.6190	0.8558	0.5935	0.5949
	Bagging	0.7279	0.8945	0.6912	0.6937	0.6224	0.8270	0.5282	0.5437

Second, for four evaluation criteria, it is easy to find that Precision can obtain the best performance. Furthermore, in comparison with the evaluation performances using the Kaggle dataset, the CUP dataset performs relatively better in terms of these metrics. Therefore, the evaluation performances of RF as a feature ranking method in the CUP dataset are better than those of other feature ranking methods, and RF is chosen as the feature ranking method for the next step of comparison. Similarly, the evaluation performances of XGBoost as a feature ranking method in the Kaggle dataset are better than those of other feature ranking methods, and XGBoost is chosen as the feature ranking method for the next step of comparison.

Finally, from the classifier perspective, the evaluation performances in terms of these metrics obtained by using an ensemble classification method (i.e., RF, XGBoost, AdaBoost, and Bagging) are better than those obtained by using an individual classification method (i.e., LDA, KNN, NB, and DT) in most circumstances. Surprisingly, the LogR surpasses other single classifiers in performance. The possible reason is that LogR is simple and robust for linear relationships, excels in binary classification, and is highly interpretable, making it more efficient and reliable than other single classifiers.

To explore which feature ranking is suitable for different datasets, the selection features are increased to the top 20%. Accordingly, the performance comparison results are presented in Table 5.

As can be seen from Table 5, four important results are summarized.

First, considering various feature ranking methods, these methods were able to improve the performance of the classification methods. Notably, AdaBoost achieved the best results in ACC, G-mean, and AUC. This can be attributed to two-fold. On the one hand, AdaBoost selected more relevant features than RF and XGBoost. On the other hand, these significant features help avoid overfitting, thus improving predictive performance.

Second, across four evaluation criteria, the LogR and XGBoost classifiers demonstrate the highest performance in ACC, Precision, G-mean, and AUC when employing various feature ranking methods. The possible reason is when AdaBoost is used as the feature ranking method, the selected top 20% of features are more relevant to the classification label. In particular, XGBoost classifier can obtain better performance in terms of the aggregative metrics, G-mean and AUC.

Third, considering the classifiers, the evaluation performance based on these metrics achieved through ensemble classification methods, such as RF, XGBoost, AdaBoost, and Bagging, is typically superior to that obtained from individual classification methods like LDA, KNN, NB, and DT. Notably, both LogR and NB outperform other individual classifiers in terms of performance. This implies that LogR and NB are simple and effective in handling different data distributions, and can obtain the robust performance with small to medium-sized datasets, making them highly effective and reliable in various scenarios.

Finally, in both the CUP and Kaggle datasets, ensemble classifiers consistently outperform individual methods across most metrics. The CUP dataset demonstrates stronger overall performance, leading to the selection of AdaBoost as the feature ranking method. Meanwhile, RF is chosen for the Kaggle dataset due to its superior evaluation results.

At this time, no consistent conclusions have been reached in the two real-world datasets. Therefore, the top 30% of features need to be further selected for the experimental analysis, and the corresponding performance comparison results are reported in Table 6.

Table 5. The top 20% features selected by the tree-based feature importance ranking methods

FS Methods	Classification Methods	The CUP dataset				The Kaggle dataset			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
AdaBoost	LDA	0.6559	0.9179	0.6951	0.6988	0.4305	0.8935	0.5253	0.5852
	LogR	0.6025	0.9184	0.6651	0.6764	0.2959	**0.9363**	0.3486	0.5464
	KNN	0.6239	0.9065	0.6666	0.6712	0.3769	0.8069	0.4568	0.5053
	NB	0.4877	0.9185	0.5789	0.6265	0.4565	0.8666	0.5388	0.5725
	DT	0.6989	0.8860	0.6674	0.6694	0.5612	0.8238	0.5387	0.5402
	RF	0.7139	0.9189	0.7250	0.7254	0.5719	0.8491	0.5800	0.5806
	XGBoost	**0.7369**	0.9110	0.7229	0.7234	0.6441	0.8413	0.5722	0.5824
	AdaBoost	0.7206	0.9184	**0.7273**	**0.7275**	0.5817	0.8499	0.5831	0.5839
	Bagging	0.7233	0.8985	0.6970	0.6984	0.6128	0.8290	0.5467	0.5554
RF	LDA	0.6350	0.9207	0.6863	0.6931	0.4394	0.9058	0.5379	0.5962
	LogR	0.6002	**0.9264**	0.6698	0.6840	0.2946	0.9192	0.3574	0.5431
	KNN	0.5998	0.9035	0.6509	0.6580	0.4100	0.8128	0.4808	0.5067
	NB	0.7146	0.8969	0.6910	0.6924	0.4336	0.8801	0.5259	0.5744
	DT	0.6862	0.8883	0.6679	0.6687	0.5622	0.8248	0.5311	0.5336
	RF	0.7107	0.9179	0.7222	0.7226	0.5664	0.8677	0.5968	**0.5997**
	XGBoost	0.7330	0.9064	0.7138	0.7146	**0.6549**	0.8578	**0.5995**	0.6057
	AdaBoost	0.7144	0.9198	0.7264	0.7268	0.5834	0.8608	0.5927	0.5937
	Bagging	0.7127	0.9006	0.6969	0.6975	0.6127	0.8395	0.5594	0.5653
XGBoost	LDA	0.6471	0.9182	0.6904	0.6955	0.4337	0.9033	0.5301	0.5933
	LogR	0.6107	0.9180	0.6696	0.6795	0.3094	0.9356	0.3759	0.5533
	KNN	0.6123	0.9053	0.6591	0.6651	0.3781	0.8037	0.4572	0.5023
	NB	0.7026	0.8990	0.6901	0.6911	0.4882	0.8676	0.5602	0.5822
	DT	0.7013	0.8857	0.6675	0.6697	0.5681	0.8199	0.5312	0.5346
	RF	0.7218	0.9164	0.7250	0.7251	0.5694	0.8546	0.5865	0.5878
	XGBoost	0.7366	0.9092	0.7199	0.7205	0.6509	0.8438	0.5769	0.5879
	AdaBoost	0.7154	0.9176	0.7239	0.7242	0.6050	0.8499	0.5879	0.5890
	Bagging	0.7306	0.9007	0.7032	0.7047	0.6290	0.8328	0.5538	0.5646

As can be seen from Table 6, the following findings can be drawn from Table 6.

First of all, from the viewpoint of various feature ranking methods, these methods are able to improve the performance of the model. In particular, it is easy to find that AdaBoost as a feature ranking method can obtain the best performance in terms of Precision, G-mean, and AUC. The underlying reasons for this phenomenon are identical to those detailed in Table 4.

Second, across four evaluation criteria, it is clear that the highest performance is achieved by utilizing the AdaBoost-based feature ranking method among various ranking methods. However, the best evaluation performances of ACC, G-mean, and AUC could be obtained when XGBoost is used as the feature ranking method in the Kaggle dataset, the possible reason is the proficiency of XGBoost in handling high-dimensional data, effective feature utilization, intricate interaction capture, and the employment of strong regularization to prevent from overfitting.

Table 6. The top 30% features selected by the tree-based feature importance ranking methods

FS Methods	Classification Methods	The CUP dataset				The Kaggle dataset			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
AdaBoost	LDA	0.6282	0.9215	0.6830	0.6911	0.4282	0.8849	0.5211	0.5782
	LogR	0.6029	0.9223	0.6685	0.6808	0.3100	**0.9351**	0.3728	0.5522
	KNN	0.6136	0.9106	0.6649	0.6720	0.3688	0.8069	0.4502	0.5049
	NB	0.4726	0.9010	0.5514	0.6074	0.4521	0.8659	0.5353	0.5705
	DT	0.6777	0.8847	0.6594	0.6602	0.5548	0.8243	0.5393	0.5404
	RF	0.7051	**0.9246**	**0.7282**	**0.7294**	0.5620	0.8517	0.5810	0.5822
	XGBoost	0.7186	0.9136	0.7192	0.7195	0.6305	0.8414	0.5727	0.5800
	AdaBoost	0.7186	0.9148	0.7209	0.7213	0.5766	0.8513	0.5834	0.5845
	Bagging	0.7153	0.9022	0.7004	0.7010	0.6074	0.8313	0.5523	0.5587
RF	LDA	0.6272	0.9236	0.6842	0.6930	0.4326	0.8814	0.5240	0.5775
	LogR	0.5934	0.9249	0.6642	0.6793	0.3085	0.9312	0.3686	0.5512
	KNN	0.5912	0.9117	0.6530	0.6640	0.3827	0.8065	0.4613	0.5058
	NB	0.4344	0.8815	0.5155	0.5816	0.4270	0.8617	0.5144	0.5601
	DT	0.6775	0.8872	0.6636	0.6641	0.5630	0.8230	0.5377	0.5394
	RF	0.7050	0.9178	0.7194	0.7199	0.5587	0.8503	0.5778	0.5793
	XGBoost	0.7265	0.9100	0.7172	0.7175	0.6442	0.8457	0.5827	0.5905
	AdaBoost	0.7054	0.9185	0.7204	0.7210	0.5712	0.8481	0.5787	0.5793
	Bagging	0.7072	0.9003	0.6947	0.6951	0.6132	0.8327	0.5550	0.5620
XGBoost	LDA	0.6427	0.9193	0.6893	0.6949	0.4671	0.8909	0.5560	**0.5966**
	LogR	0.6091	0.9176	0.6685	0.6782	0.3397	0.9231	0.4212	0.5629
	KNN	0.6157	0.9078	0.6635	0.6696	0.3896	0.8056	0.4659	0.5048
	NB	0.5664	0.9037	0.6269	0.6451	0.4867	0.8659	0.5585	0.5801
	DT	0.6909	0.8886	0.6699	0.6710	0.5694	0.8260	0.5425	0.5446
	RF	0.7104	0.9176	0.7216	0.7220	0.5741	0.8514	**0.5841**	0.5845
	XGBoost	**0.7261**	0.9105	0.7179	0.7181	**0.6562**	0.8387	0.5651	0.5799
	AdaBoost	0.7138	0.9180	0.7237	0.7240	0.6066	0.8440	0.5781	0.5801
	Bagging	0.7172	0.9029	0.7025	0.7030	0.6275	0.8297	0.5467	0.5588

Third, the classifier performance in Table 6 indicates that using AdaBoost as the feature ranking method results in superior Precision, G-mean, and AUC for the RF, LogR, and NB classifiers. This improvement is attributed to AdaBoost's ability to prioritize key features, thereby enhancing the accuracy and robustness of these algorithms. Conversely, when XGBoost ranks features, both XGBoost and RF excel in ACC and G-mean, while LDA, DT, and KNN exhibit moderate improvements, and bagging performs the worst. XGBoost's effectiveness in managing complex feature interactions and regularization enhances the performance of both XGBoost and RF. Simple models like LDA, DT, and KNN benefit less from this method, while bagging struggles with nuanced feature importance. Overall, ensemble models such as RF, XGBoost, and AdaBoost, along with LogR, demonstrate strong classification capabilities. The slightly inferior performance of bagging may result from its inability to capture informative features in high-dimensional spaces and its susceptibility to local minima.

Finally, for both the CUP and Kaggle datasets, the evaluation performance for each metric achieved by individual classification methods is typically lower than that obtained through ensemble classification methods. The possible reason is that the ensemble classification methods combine the strengths of multiple classifiers, reducing errors and improving overall prediction robustness. Also, in comparison with the evaluation performances using the Kaggle dataset, the CUP dataset performs relatively better in terms of these metrics. This superiority is likely attributed to the higher discriminative power of its features and their better alignment with the employed classification methods, as indicated by consistently higher values in Precision and AUC.

Therefore, the evaluation performances of AdaBoost as a feature ranking method in the CUP dataset are better than those of other feature ranking methods, and AdaBoost is selected as the feature ranking method in the CUP dataset for the next step of comparison. Similarly, the evaluation performances of XGBoost as a feature ranking method in Kaggle dataset is better than those of other feature ranking methods, and XGBoost is selected as the feature ranking method in the Kaggle dataset for the next step of comparison.

Based on the results presented in Tables 4–6, the optimal evaluation metrics were selected to determine the most effective feature ranking methods for two real-world credit datasets. The outcomes for the CUP and Kaggle datasets are illustrated in Figures 4 and 5, respectively, where the horizontal axis represents the classification methods, and the vertical axis illustrates the scores for each evaluation metric. From Figure 4, it is found that the first 9 classifiers utilize RF to select the top 10% of features, the subsequent 9 classifiers employ AdaBoost to select the top 20%, and the final 9 classifiers use AdaBoost to select the top 30%. Meantime, AdaBoost, selecting the top 30% of features, outperforms the other methods across all metrics, establishing it as the preferred method for the CUP dataset when compared to traditional FS methods.

Similarly, in Figure 5, the horizontal and vertical axes represent the classification methods and evaluation metric scores, respectively. XGBoost, selecting the top 10% of features, surpasses RF (selecting 20%) and XGBoost (selecting 30%) in all metrics. Consequently, XGBoost, selecting the top 10% of features, is the preferred method for the Kaggle dataset relative to traditional FS methods.

5.2.2. Comparison of FS techniques

Based on the results presented in Section 5.2.1 and the traits of various high-dimensional credit datasets, optimal FS methods were selected for comparison with traditional methods. The AdaBoost method was applied to the CUP dataset, selecting the top 30% of features, while the XGBoost method was utilized for the Kaggle dataset, selecting the top 10% features. The results of the proposed CBTFS method are shown in Table 7 alongside traditional FS methods, comparing four evaluation metrics across the two datasets. The best result for each metric is highlighted in bold, with rankings indicated in parentheses.

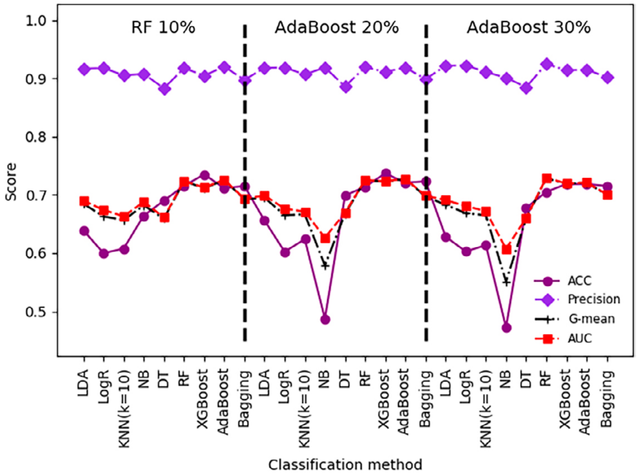

Figure 4. Ranking the importance of the top features for the CUP dataset

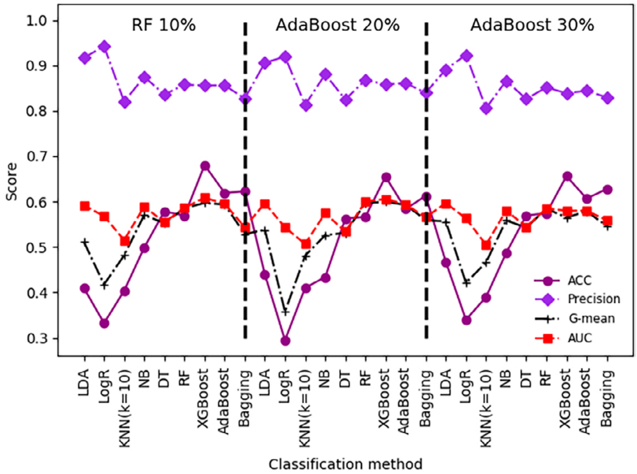

Figure 5. Ranking the importance of the for the Kaggle dataset

As can be seen from Table 7, in the CUP dataset, the ACC of the CBTFS model is weaker than that of traditional FS methods and those without FS. This may result from overfitting or inadequate FS that fails to capture relevant data features, as well as performance bias towards the majority class due to dataset imbalance. Conversely, the Precision, G-mean, and AUC metrics of the CBTFS are superior to those of traditional FS methods, likely because the hybrid CBTFS model effectively identifies and retains the most informative features, thereby enhancing its ability to correctly identify positive instances. Notably, the top three best results, particularly for G-mean and AUC, are attributed to the CBTFS model. Similarly, in the Kaggle dataset, the CBTFS model demonstrates poorer ACC but better Precision, G-mean, and AUC compared to traditional FS methods. This may be attributed to the CBTFS model's focus on prioritizing relevant features, which can result in a higher number of misclassifications in the overall dataset, particularly among negative instances.

Table 7. Comparison of the proposed CBTFS model and other FS methods

FS methods	Classification Method	The CUP dataset (AdaBoost 30%)				The Kaggle dataset (XGBoost 10%)			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
Proposed Method (CBTFS)	LDA	0.6282	**0.9215(3)**	0.6830	0.6911	0.4093	**0.9167(2)**	0.5110	0.5914
	LogR	0.6029	**0.9223(2)**	0.6685	0.6808	0.3322	**0.9420(1)**	0.4174	0.5680
	KNN	0.6136	0.9106	0.6649	0.6720	0.4036	0.8201	0.4815	0.5150
	NB	0.4726	0.9010	0.5514	0.6074	0.4989	**0.8747(3)**	0.5703	0.5892
	DT	0.6777	0.8847	0.6594	0.6602	0.5782	0.8356	0.5525	0.5541
	RF	0.7051	**0.9246(1)**	**0.7282(1)**	**0.7294(1)**	0.5692	0.8575	0.5854	0.5862
	XGBoost	0.7186	0.9136	**0.7192(3)**	**0.7195(3)**	0.6803	0.8557	**0.5969(1)**	**0.6085(2)**
	AdaBoost	0.7186	0.9148	**0.7209(2)**	**0.7213(2)**	0.6190	0.8558	**0.5935(2)**	0.5949
	Bagging	0.7153	0.9022	0.7004	0.7010	0.6224	0.8270	0.5282	0.5437
VAR	LDA	0.8086	0.8365	0.4685	0.5903	**0.8324(1)**	0.8422	0.4875	**0.6095(1)**
	LogR	**0.8126(2)**	0.8332	0.4422	0.5821	0.7973	0.8162	0.3090	0.5328
	KNN	0.8021	0.8180	0.3212	0.5385	0.7967	0.8059	0.1315	0.5015
	NB	0.2794	0.8564	0.3390	0.5193	0.5886	0.8563	0.5861	0.5888
	DT	0.7293	0.8388	0.5324	0.5840	0.6898	0.8172	0.4589	0.5294
	RF	0.8073	0.8245	0.3786	0.5576	0.8026	0.8059	0.0776	0.5016
	XGBoost	0.8067	0.8417	0.5028	0.6035	0.7930	0.8145	0.2973	0.5275
	AdaBoost	0.8060	0.8371	0.4751	0.5917	0.7982	0.8137	0.2760	0.5256
	Bagging	0.7964	0.8284	0.4251	0.5673	0.7881	0.8089	0.2312	0.5107
Relief	LDA	0.8111	0.8350	0.4574	0.5868	**0.8060(3)**	0.8173	0.2953	0.5363
	LogR	0.8094	0.8284	0.4085	0.5686	0.8019	0.8108	0.2187	0.5168
	KNN	0.7882	0.8303	0.4453	0.5712	0.7764	0.8115	0.2979	0.5183
	NB	0.7133	0.8827	0.6611	0.6667	0.5783	0.8522	0.5508	0.5746
	DT	0.7301	0.8366	0.5236	0.5793	0.6955	0.8208	0.4720	0.5382
	RF	0.8078	0.8261	0.3901	0.5620	0.8044	0.8064	0.0979	0.5033
	XGBoost	0.8038	0.8386	0.4863	0.5952	0.7921	0.8144	0.2969	0.5271
	AdaBoost	0.8038	0.8328	0.4490	0.5800	0.7999	0.8109	0.2268	0.5171
	Bagging	0.7944	0.8284	0.4273	0.5671	0.7908	0.8200	0.2417	0.5141
ReliefF	LDA	**0.8172(1)**	0.8407	0.4966	0.6061	0.7858	0.8390	0.4726	0.5948
	LogR	0.8156	0.8339	0.4536	0.5881	0.7916	0.8103	0.2959	0.5284
	KNN	0.8074	0.8190	0.3348	0.5456	0.7925	0.8007	0.1125	0.4997
	NB	0.7391	0.8807	0.6648	0.6736	0.6147	0.8352	0.5606	0.5666
	DT	0.6255	0.8229	0.5207	0.5402	0.6547	0.8103	0.4729	0.5223
	RF	0.8043	0.8161	0.3056	0.5369	0.7989	0.8019	0.1057	0.5033
	XGBoost	0.8001	0.8245	0.3906	0.5600	0.7900	0.8085	0.2759	0.5230
	AdaBoost	0.7998	0.8111	0.2383	0.5216	0.7960	0.8088	0.2646	0.5243
	Bagging	0.7273	0.8168	0.4280	0.5345	0.7711	0.8060	0.2898	0.5152
mRMR	LDA	0.8039	0.8061	0.0766	0.5021	0.4706	0.8491	0.5381	0.5565
	LogR	0.8038	0.8060	0.0737	0.5019	0.4702	0.8490	0.5379	0.5563
	KNN	0.7937	0.8096	0.2184	0.5129	0.5519	0.8235	0.5301	0.5320
	NB	0.7844	0.8181	0.3324	0.5364	0.4510	0.8494	0.5267	0.5527
	DT	0.6821	0.8226	0.4908	0.5422	0.4038	0.8201	0.4645	0.5145
	RF	0.7791	0.8156	0.3258	0.5297	0.4967	0.8233	0.5213	0.5264
	XGBoost	0.7829	0.8097	0.2521	0.5129	0.3472	0.8400	0.3882	0.5207
	AdaBoost	0.8038	0.8053	0.0233	0.4997	0.4787	0.8340	0.5285	0.5384
	Bagging	0.7528	0.8144	0.3673	0.5254	0.4689	0.8263	0.5059	0.5280
Original	LDA	0.8086	0.8358	0.4645	0.5886	**0.8323(2)**	0.8418	0.4845	**0.6083(3)**
	LogR	**0.8112(2)**	0.8319	0.4345	0.5786	0.7979	0.8166	0.3122	0.5339
	KNN	0.8039	0.8199	0.3378	0.5440	0.7979	0.8061	0.1271	0.5021
	NB	0.2811	0.8516	0.3413	0.5185	0.5897	0.8573	**0.5898(3)**	0.5911
	DT	0.7378	0.8438	0.5492	0.5967	0.6953	0.8187	0.4626	0.5335
	RF	0.8089	0.8255	0.3847	0.5606	0.8021	0.8058	0.0813	0.5013
	XGBoost	0.8039	0.8386	0.4870	0.5953	0.7909	0.8135	0.2903	0.5246
	AdaBoost	0.8073	0.8380	0.4800	0.5942	0.7979	0.8133	0.2699	0.5241
	Bagging	0.8025	0.8312	0.4389	0.5756	0.7880	0.8091	0.2356	0.5113

To further verify the effectiveness of the proposed hybrid CBTFS method, this paper extends its analysis beyond the comparison of the four typical filtered methods listed in Table 7. Additionally, a comparative evaluation is conducted by using three commonly utilized intelligent algorithms, genetic algorithm (GA), whale optimization algorithm (WOA), and particle swarm optimization (PSO). At the same time, two real-world credit datasets including CUP credit dataset and the Kaggle credit dataset are used. The results of the five FS methods in two datasets are listed in Table 8, with the best results in bold to represent the optimization impact of each method.

Table 8. Comparison of the proposed CBTFS model and other evolutionary algorithm FS methods

FS methods	Classification Methods	The CUP dataset				The Kaggle dataset			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
Proposed Method (CBTFS)	LDA	0.6282	**0.9215(3)**	0.6830	0.6911	0.4093	**0.9167(3)**	0.5110	**0.5914(3)**
	LogR	0.6029	**0.9223(2)**	0.6685	0.6808	0.3322	**0.9420(2)**	0.4174	0.5680
	KNN	0.6136	0.9106	0.6649	0.6720	0.4036	0.8201	0.4815	0.5150
	DT	0.6777	0.8847	0.6594	0.6602	0.5782	0.8356	0.5525	0.5541
	RF	0.7051	**0.9246(1)**	**0.7282(1)**	**0.7294(1)**	0.5692	0.8575	0.5854	0.5862
	XGBoost	0.7186	0.9136	**0.7192(3)**	**0.7195(3)**	0.6803	0.8557	**0.5969(1)**	**0.6085(1)**
	AdaBoost	0.7186	0.9148	**0.7209(2)**	**0.7213(2)**	0.6190	0.8558	**0.5935(2)**	**0.5949(2)**
	Bagging	0.7153	0.9022	0.7004	0.7010	0.6224	0.8270	0.5282	0.5437
	Average	**0.6725**	**0.9117**	**0.6930**	**0.6969**	**0.5267**	0.8638	**0.5333**	**0.5702**
GA	LDA	0.4738	0.8632	0.5230	0.5658	0.4770	0.9060	0.4941	0.4971
	LogR	0.7566	0.8375	0.5087	0.5847	0.6438	0.9122	0.4967	0.5197
	KNN	**0.8008(2)**	0.8048	0.0279	0.4982	0.9072	0.9072	0.0000	0.5000
	DT	0.4404	0.8083	0.2795	0.4932	0.3442	0.8879	0.3121	0.4747
	RF	0.5655	0.8214	0.4408	0.5183	0.1203	0.9002	0.1270	0.4980
	XGBoost	0.6842	0.8193	0.3715	0.5250	0.1960	0.9162	0.3186	0.5091
	AdaBoost	**0.8053(1)**	0.8053	0.0000	0.5000	0.9037	0.9075	0.0457	0.5020
	Bagging	0.4556	0.8155	0.3781	0.5168	0.3147	0.9029	0.3336	0.5097
	Average	0.6227	0.8219	0.3161	0.5252	0.4883	0.9050	0.2659	0.5012
WOA	LDA	0.4719	0.8492	0.5287	0.5526	0.5753	**0.9489(1)**	0.5187	0.6194
	LogR	0.7846	0.8171	0.3317	0.5343	0.6382	0.9134	0.5054	0.5244
	KNN	0.7720	0.8078	0.1214	0.5053	0.9072	0.9072	0.0000	0.5000
	DT	0.5024	0.7928	0.3302	0.4710	0.2976	0.8821	0.3305	0.5141
	RF	0.4186	0.8350	0.4325	0.5137	0.1224	0.9126	0.1653	0.5001
	XGBoost	0.5681	0.8277	0.4447	0.5300	0.3269	0.9071	0.4225	0.5059
	AdaBoost	**0.8053(1)**	0.8053	0.0000	0.5000	0.9072	0.9072	0.0000	0.5000
	Bagging	0.5408	0.7963	0.3734	0.4724	0.2163	0.8874	0.3325	0.4907
	Average	0.6079	0.8164	0.3203	0.5099	0.4988	**0.9082**	0.2843	0.5193
PSO	LDA	0.6287	0.8287	0.4960	0.5487	0.1987	0.9071	0.3168	0.4973
	LogR	**0.7738(3)**	0.8120	0.2978	0.5193	0.7588	0.9105	0.4208	0.5160
	KNN	0.7486	0.8135	0.3010	0.5211	0.9072	0.9072	0.0000	0.5000
	DT	0.4148	0.7302	0.3661	0.4840	0.3707	0.9128	0.4185	0.5044
	RF	0.4495	0.7660	0.4354	0.5353	0.1040	0.9218	0.1072	0.5001
	XGBoost	0.6389	0.8258	0.3851	0.5276	0.5066	0.9129	0.4069	0.5094
	AdaBoost	0.6868	0.9114	0.7015	0.7023	0.9064	0.9073	0.0370	0.5007
	Bagging	0.6995	0.9016	0.6927	0.6930	0.1782	0.8798	0.3023	0.4845
	Average	0.6300	0.8236	0.4594	0.5664	0.4913	0.9074	0.2511	0.5015

As can be seen from Table 8, three interesting results can be found below.

First, the proposed hybrid CBTFS methodology consistently outperforms other intelligent FS methods, including GA, WOA, and PSO, across different metrics such as ACC, Precision, G-mean, and AUC. Mean value comparisons further demonstrate the superiority of the CBTFS method over other intelligent algorithms. The possible reason is that the CBTFS method enhances the accuracy and stability of FS by integrating multiple strategies, leading to superior performance across various metrics.

Second, the performance of the RF, AdaBoost, and XGBoost classifiers within the CBTFS framework generally exceeds that of other FS methods. The possible reason is that these ensemble classifiers inherently possess strong generalization capabilities and can capture complex patterns. The CBTFS framework further enhances their performance by efficiently selecting some typical features that excel across various datasets and metrics.

Finally, although the CBTFS method excels in most metrics, its ACC performance is relatively lower, likely due to data imbalance that results in the misclassification of some creditworthy samples. Nevertheless, the CBTFS method averages better than other intelligent FS algorithms.

Overall, the proposed hybrid CBTFS method outperforms other FS methods across various evaluation metrics. In both the CUP and Kaggle datasets, the ensemble classifier based on CBTFS achieves superior results compared to traditional FS methods in terms of ACC, G-mean, and AUC. However, the individual classifier utilizing CBTFS demonstrates better Precision than traditional FS methods. This phenomenon can be attributed to two reasons. First, the hybrid CBTFS method employs an advanced FS process that captures relevant data attributes, thereby enhancing model accuracy and robustness. Second, while the ensemble classifier excels in ACC, G-mean, and AUC by leveraging multiple models, individual classifiers achieve higher Precision by focusing on accurately identifying specific instances. This underscores the effectiveness of the CBTFS method in high-dimensional credit datasets.

5.2.3. Comparison of ablation experiments

A well-designed credit risk classification model must have components that are both essential and compatible. To further validate the effectiveness of the proposed CBTFS methodology, ablation experiments are conducted. These experiments assess the key components to verify the structural integrity of the CBTFS method. Additionally, to present the experimental results clearly, Table 9 displays the average performance of the eight classifiers, with the best results highlighted in bold to emphasize the optimization effects of each method. For detailed performance metrics of each classifier, please refer in Appendix.

As can be seen from Table 9, some interesting results can be found below.

First, from an overall perspective, the proposed CBTFS method is generally considered to be reasonable since the exclusion of any module can adversely affect the performance. In most cases, the CBTFS method proposed in this paper achieves best performance, which further highlights that the CBTFS method proposed in this paper is effective.

Second, the poorest performance occurs when only IMST (*i.e.*, w/o AdaBoost, RF, and XGBoost) is used for FS, excluding AdaBoost, RF, and XGBoost. This may be due to IMST's inability to adequately account for feature interactions, leading to unresolved feature redundancy issues.

Table 9. Comparison of ablation experiments

FS methods	The CUP dataset				The Kaggle dataset			
	ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
Proposed CBTFS	0.6725	**0.9117**	**0.6930**	**0.6969**	0.5267	0.8638	0.5333	0.5702
w/o AdaBoost, RF, and XGBoost	0.4170	0.7739	0.3524	0.4983	**0.5652**	0.8147	0.3104	0.5066
w/o IMST, RF, and XGBoost	0.6520	0.9062	0.6752	0.6801	0.4834	0.8882	0.5542	**0.5907**
w/o IMST, AdaBoost, and XGBoost	0.6551	0.9000	0.6693	0.6730	0.4312	**0.8981**	0.5159	0.5833
w/o IMST, AdaBoost, and RF	**0.6878**	0.9008	0.6849	0.6867	0.5117	0.8800	**0.5652**	0.5167

Third, when only AdaBoost (*i.e.*, w/o IMST, RF, and XGBoost) is used, excluding IMST, RF, and XGBoost, the performance surpasses that of using only RF and IMST. That is, AdaBoost effectively addresses feature importance and enhances weak classifiers through a weighting mechanism. Although performance decreases with only XGBoost (*i.e.*, w/o IMST, AdaBoost, and RF), it provides the best results in the ablation experiments. This is slightly lower than CBTFS, likely due to the absence of diversity and complementary information from other methods.

Finally, CBTFS outperforms on the CUP dataset compared to the Kaggle dataset, possibly because the features of CUP dataset are more compatible than those of Kaggle dataset within the CBTFS framework. Meanwhile, the performance of the CBTFS method is observed to be slightly lower than that of other post-ablation methods in the ablation experiments. This may be attributed to the higher levels of noise and redundancy among features in the Kaggle dataset, which hinder the CBTFS model's ability to effectively extract useful information. In contrast, the CUP dataset exhibits greater compatibility among features, enabling CBTFS to leverage its strengths and enhance performance. Overall, the ablation experiments further confirm that the proposed CBTFS method can achieve better results than other alternative methods.

5.2.4. Statistical test results on the two datasets

In this Section, a significance test was conducted to compare the performance differences among various models. The paired sample *t*-test and Wilcoxon test were used to assess discrepancies between the CBTFS model and four traditional FS methods across ACC, Precision, G-mean, and AUC, as shown in Tables 10 and 11.

From Table 10, the CBTFS method significantly outperforms the traditional FS methods in ACC, Precision, G-mean, and AUC at the 5% significance level in the CUP dataset. In the Kaggle dataset, most metrics for CBTFS also show significant differences compared to the traditional FS methods. Notably, results for ACC and G-mean differ markedly from the mRMR FS model, likely due to data imbalance trait of Kaggle dataset. To address this issue, minority classes were augmented by using Synthetic Minority Oversampling Technique (SMOTE) and Edited Nearest Neighbor (ENN) methods. The results confirm CBTFS's substantial advantage over other methods in both datasets, with generally higher evaluation metrics. This demonstrates CBTFS's superior performance and sorting abilities with minimal impact on accuracy.

Table 10. Results of paired *t*-test (*p*-value) for comparison of five evaluation metrics

Metrics	CBTFS vs VAR	CBTFS vs Relief	CBTFS vs ReliefF	CBTFS vs mRMR
ACC	–2.2210*	–6.0010***	–3.5050***	–3.8700***
Precision	11.6700***	8.7340***	9.2110***	16.5720***
G-mean	10.5990***	4.7390***	4.2950***	7.3610***
AUC	11.9790***	4.6390***	4.4490***	10.6880***
Metrics	CBTFS vs VAR	CBTFS vs Relief	CBTFS vs ReliefF	CBTFS vs mRMR
ACC	–4.9390***	–5.0130***	–4.9110***	1.2090
Precision	3.7110***	3.2830***	4.3860***	2.8720**
G-mean	3.7810***	5.0010***	4.0050***	1.0530
AUC	3.0010*	4.1890***	3.9800***	3.5450***

Note: * represents significance at the 10% level. ** represents significance at the 5% level. *** represents significance at the 1% level.

Similarly, the Wilcoxon test is a nonparametric statistical method that effectively compares the median differences between two related sample groups, making it particularly suitable for data that do not follow a normal distribution. Therefore, the Wilcoxon test is employed in this paper to further validate the usefulness and stability of the CBTFS method. The significance of this test for the performance metrics of the CBTFS models on the CUP and Kaggle datasets is presented in Table 11.

The null hypothesis for each metric, as shown in Table 11, is that no performance difference exists between the two methods. Significance levels are established at 1%, 5%, and 10%. The null hypothesis is rejected when the *p*-values fall below these thresholds. It is shown in Table 11 that most *p*-values for ACC, Precision, G-mean, and AUC are below 1% (*i.e.*, the presence of 3 stars *** significantly), with only a few insignificant *p*-values indicated in bold. This demonstrates that significant differences exist between the CBTFS method and the four traditional FS methods, demonstrating that the CBTFS method outperforms these alternatives in high-dimensional credit datasets.

5.3. Summarizations

By comparing the results of all models in Tables 3–11 and Figures 2–5, the proposed hybrid CBTFS technique can effectively solve the high dimensionality issue in credit risk dataset and improve AUC, ACC, Precision, and G-mean of the credit classification models, therefore minimizing possible loss for financial institutions.

In terms of the empirical results of the above two experiments, three main conclusions can be drawn below.

(1) The proposed hybrid CBTFS method can improve classification performance greatly in both credit datasets. There are some differences in the findings obtained from these two credit datasets, which may be mainly due to the inconsistency in the structural characteristics of these two datasets. Generally, the overall performance of the proposed hybrid CBTFS method is better than that of other traditional FS methods, as shown in Table 5. It can also address the high dimensionality issue well in credit risk assessment. However, by comparing the predictive results when the top 10 %, 20% and 30% features are selected, the principle of "the more features, the better performance" does not hold.

(2) The proposed hybrid CBTFS method is an effective method to enhance general performance and address the high-dimensional data issue. The comparison between the proposed hybrid CBTFS method and other widely used intelligent algorithms demonstrates its superior performance relative to alternative intelligent methods. Furthermore, FS is necessary after feature construction to improve the classification performance when facing the high-dimensional data issue.

(3) Compared with all the different single classification methods (linear and non-linear) and ensemble classification methods, the proposed hybrid CBTFS method with linear classifiers and ensemble classifiers perform the best when hybrid CBTFS is applied to FS. For financial institutions, this means that the risk of financial losses can be minimized.

Table 11. Results of non-parametric Wilcoxon test for evaluation metrics on two datasets

Metrics	FS	The CUP dataset					The Kaggle dataset				
		VAR	Relief	ReliefF	mRMR	Proposed	VAR	Relief	ReliefF	mRMR	Proposed
ACC	VAR	–	**–0.6520**	**–0.3560**	**–1.5990**	–1.7180*	–	**–0.2960**	–1.8360*	–2.6660***	–2.6680***
	Relief	–	–	**–0.0590**	**–1.2600**	–2.668***	–	–	**–1.1250**	–2.6660***	–2.6680***
	ReliefF	–	–	–	**–0.2960**	–2.4290***	–	–	–	–2.6660***	–2.6680***
	mRMR	–	–	–	–	–2.6660***	–	–	–	–	–1.2440
Precision	FS	VAR	Relief	ReliefF	mRMR	Proposed	VAR	Relief	ReliefF	mRMR	Proposed
	VAR	–	**–0.1400**	**–1.0070**	–2.6660***	–2.6660***	–	**–0.1780**	–2.6660***	–2.3740**	–2.6680***
	Relief	–	–	–2.0730**	–2.6700***	–2.6660***	–	–	**–1.5990**	–2.3100**	–2.6660***
	ReliefF	–	–	–	–2.6660***	–2.6660***	–	–	–	–2.6660***	–2.6660***
	mRMR	–	–	–	–	–2.6660***	–	–	–	–	–2.4290**
G-mean	FS	VAR	Relief	ReliefF	mRMR	Proposed	VAR	Relief	ReliefF	mRMR	Proposed
	VAR	–	**–0.0590**	**–0.1780**	–2.6660***	–2.6660***	–	**–0.6520**	**–0.2960**	–2.310**	–2.5470**
	Relief	–	–	**–1.1250**	–2.6660***	–2.5470**	–	–	**–1.1250**	–2.310**	–2.6660***
	ReliefF	–	–	–	–2.5470**	–2.5470**	–	–	–	–2.310**	–2.6660***
	mRMR	–	–	–	–	–2.6660***	–	–	–	–	–1.0070
AUC	FS	VAR	Relief	ReliefF	mRMR	Proposed	VAR	Relief	ReliefF	mRMR	Proposed
	VAR	–	**–0.2960**	**–0.8890**	–25470**	–2.6660***	–	**–0.5330**	–1.7790*	**–0.1780**	–2.3100**
	Relief	–	–	–1.9550*	–2.6660***	–2.5470**	–	–	**–0.1400**	**–0.8890**	–2.5470**
	ReliefF	–	–	–	–2.5470**	–2.5470**	–	–	–	**–0.7700**	–2.5470**
	mRMR	–	–	–	–	–2.6660***	–	–	–	–	–2.3100**

Note: * represents significance at the 10% level. ** represents significance at the 5% level. *** represents significance at the 1% level.

6. Conclusions and future directions

In this paper, a hybrid clustering and boosting tree-based feature selection (CBTFS) method is proposed for high-dimensional credit risk classification. Given the high-dimensional data features, an improved minimum spanning tree (IMST) model is first employed to remove redundant and irrelevant features. Subsequently, three embedded feature selection methods – RF, XGBoost, and AdaBoost – are used to further improve the efficiency of feature ranking. Thus, a hybrid CBTFS method is proposed to address the high-dimensional problem of credit datasets.

For validation and comparison, two credit datasets and three types of classifiers are used to test the effectiveness of the proposed method. The results reported in both experiments clearly show that the hybrid CBTFS method can improve classification performance and significantly outperforms the other algorithms listed in this study. The empirical results indicate that the proposed hybrid method can effectively solve the high-dimensional feature dataset problem in credit risk assessment, suggesting that the proposed CBTFS method provides a promising solution for high-dimensional credit risk assessment.

Moreover, the study provides valuable insights for credit risk management, particularly in addressing high-dimensional data challenges in financial institutions through the innovative CBTFS method. By prioritizing relevant features and minimizing redundancy, the CBTFS method improves the performance of credit risk assessments, reduces defaults, and enhances financial stability. Its scalability makes it especially valuable for global institutions handling diverse high-dimensional datasets. At the same time, the study also emphasizes the importance of continuously refining feature selection methods to keep pace with evolving credit risks. Integrating advanced techniques like CBTFS into risk management frameworks can help anticipate threats and improve data processing. The adoption of CBTFS strengthens credit risk models, enhances risk-adjusted returns, and promotes equitable lending practices. In summary, the CBTFS method offers both technical and strategic advantages, reinforcing decision-making and financial resilience in credit risk management.

Although the CBTFS method effectively addresses feature redundancy and irrelevance, but several aspects require further research. First, some novel techniques should be developed to handle high-dimensional datasets with limited sample sizes, ensuring robust performance despite small data volumes. Second, enhancing model predictive power through data-trait-driven modeling, which tailors feature selection to specific data traits, is a promising direction. Third, testing the method on more diverse real-world datasets will validate its effectiveness and robustness in various contexts. Finally, exploring CBTFS applications in different domains like fraud detection, peer-to-peer lending, and credit rating will demonstrate its versatility and broader impact. In summary, CBTFS represents a significant advancement in credit risk classification with high dimensionality, offering a robust solution for financial analytics and paving the way for future research and applications.

Funding

This work is partially supported by grants from the Technical Field Fund of Basic Research Strengthening Program (Project no. 2021-JCJQ-JJ-0003), the Fundamental Research Special Funds for the Central Universities-Research and Innovation Fund for Doctoral Students (No. XK2090021029), the Nature Science Foundation of Heilongjiang (No. LH2022G00), and the National Natural Science Foundation of China (No. 72361014).

Disclosure statement

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.

Author contributions

The contribution of every author in the above paper is shown below:

Jianxin Zhu: conceptualization, investigation, and writing – review.

Xiong Wu: data curation, investigation, software python, visualization, validation, writing-original draft preparation.

Lean Yu: conceptualization, methodology, investigation, writing – review & editing, and Supervision.

Xiaoming Zhang: conceptualization, methodology, investigation, writing – review & editing.

References

Avuçlu, E. (2021). A new data augmentation method to use in machine learning algorithms using statistical measurements. *Measurement*, *180*, Article 109577. https://doi.org/10.1016/j.measurement.2021.109577

Baser, F., Koc, O., & Selcuk-Kestel, A. S. (2023). Credit risk evaluation using clustering based fuzzy classification method. *Expert Systems with Applications*, *223*, Article 119882. https://doi.org/10.1016/j.eswa.2023.119882

Belás, J., Smrcka, L., Gavurova, B., & Dvorsky, J. (2018). The impact of social and economic factors in the credit risk management of SME. *Technological and Economic Development of Economy*, *24*(3), 1215–1230. https://doi.org/10.3846/tede.2018.1968

Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, *40*(1), 16–28. https://doi.org/10.1016/j.compeleceng.2013.11.024

Chaudhuri, A. (2024). Search space division method for wrapper feature selection on high-dimensional data classification. *Knowledge-Based Systems*, *291*, Article 111578. https://doi.org/10.1016/j.knosys.2024.111578

Chowdhury, N. K., Kabir, M. A., Rahman, M. M., & Islam, S. M. S. (2022). Machine learning for detecting COVID-19 from cough sounds: An ensemble-based MCDM method. *Computers in Biology and Medicine*, *145*, Article 105405. https://doi.org/10.1016/j.compbiomed.2022.105405

Costea, A., Ferrara, M., & Serban, F. (2017). An integrated two-stage methodology for optimising the accuracy of performance classification models. *Technological and Economic Development of Economy*, *23*(1), 111–139. https://doi.org/10.3846/20294913.2016.1213196

Gonçalves, T. S., Ferreira, F. A., Jalali, M. S., & Meidutė-Kavaliauskienė, I. (2016). An idiosyncratic decision support system for credit risk analysis of small and medium-sized enterprises. *Technological and Economic Development of Economy*, *22*(4), 598–616. https://doi.org/10.3846/20294913.2015.1074125

Görüş, V., Bahşı, M. M., & Çevik, M. (2024). Machine learning for the prediction of problems in steel tube bending process. *Engineering Applications of Artificial Intelligence*, *133*, Article 108584. https://doi.org/10.1016/j.engappai.2024.108584

Gramegna, A., & Giudici, P. (2021). Shap and LIME: An evaluation of discriminative power in credit risk. *Frontiers in Artificial Intelligence*, *4*, Article 752558. https://doi.org/10.3389/frai.2021.752558

Gunnarsson, B. R., vanden Broucke, S., Baesens, B., Óskarsdóttir, M., & Lemahieu, W. (2021). Deep learning for credit scoring: Do or don't? *European Journal of Operational Research*, *295*(1), 292–305. https://doi.org/10.1016/j.ejor.2021.03.006

He, H., Zhang, W., & Zhang, S. (2018). A novel ensemble method for credit scoring: Adaption of different imbalance ratios. *Expert Systems with Applications*, *98*, 105–117. https://doi.org/10.1016/j.eswa.2018.01.012

Hu, Y., Zhang, Y., Gao, X., Gong, D., Song, X., Guo, Y., & Wang, J. (2023). A federated feature selection algorithm based on particle swarm optimization under privacy protection. *Knowledge-Based Systems*, *260*, Article 110122. https://doi.org/10.1016/j.knosys.2022.110122

Huang, S., Zhang, J., Yang, C., Gu, Q., Li, M., & Wang, W. (2022). The interval grey QFD method for new product development: Integrate with LDA topic model to analyze online reviews. *Engineering Applications of Artificial Intelligence*, *114*, Article 105213. https://doi.org/10.1016/j.engappai.2022.105213

Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V., Fidalgo, E., & Alegre, E. (2023). Classifying spam emails using agglomerative hierarchical clustering and a topic-based approach. *Applied Soft Computing*, *139*, Article 110226. https://doi.org/10.1016/j.asoc.2023.110226

Kozodoi, N., Lessmann, S., Papakonstantinou, K., Gatsoulis, Y., & Baesens, B. (2019). A multi-objective approach for profit-driven feature selection in credit scoring. *Decision Support Systems*, *120*, 106–117. https://doi.org/10.1016/j.dss.2019.03.011

Kuo, T., & Wang, K.-J. (2022). A hybrid k-prototypes clustering approach with improved sine-cosine algorithm for mixed-data classification. *Computers & Industrial Engineering*, *169*, Article 108164. https://doi.org/10.1016/j.cie.2022.108164

Li, H., & Wang, J. (2023). CAPKM++2.0: An upgraded version of the collaborative annealing power k-means++ clustering algorithm. *Knowledge-Based Systems*, *262*, Article 110241. https://doi.org/10.1016/j.knosys.2022.110241

Li, M., Ma, H., Lv, S., Wang, L., & Deng, S. (2024a). Enhanced NSGA-II-based feature selection method for high-dimensional classification. *Information Sciences*, *663*, Article 120269. https://doi.org/10.1016/j.ins.2024.120269

Li, Q., Zhao, S., He, T., & Wen, J. (2024b). A simple and efficient filter feature selection method via document-term matrix unitization. *Pattern Recognition Letters*, *181*, 23–29. https://doi.org/10.1016/j.patrec.2024.02.025

Liu, F., & Deng, Y. (2021). Determine the number of unknown targets in open world based on elbow method. *IEEE Transactions on Fuzzy Systems*, *29*(5), 986–995. https://doi.org/10.1109/TFUZZ.2020.2966182

Liu, H., Zhang, J., Liu, Q., & Cao, J. (2022). Minimum spanning tree based graph neural network for emotion classification using EEG. *Neural Networks*: *The Official Journal of the International Neural Network Society*, *145*, 308–318. https://doi.org/10.1016/j.neunet.2021.10.023

Liu, X, Li, Y., Dai, C., & Zhang, H. (2024). A hierarchical attention-based feature selection and fusion method for credit risk assessment. *Future Generation Computer Systems*, *160*, 537–546. https://doi.org/10.1016/j.future.2024.06.036

Macedo, F., Valadas, R., Carrasquinha, E., Oliveira, M. R., & Pacheco, A. (2022). Feature selection using decomposed mutual information maximization. *Neurocomputing*, *513*, 215–232. https://doi.org/10.1016/j.neucom.2022.09.101

Maldonado, S., Pérez, J., & Bravo, C. (2017). Cost-based feature selection for support vector machines: An application in credit scoring. *European Journal of Operational Research*, *261*(2), 656–665. https://doi.org/10.1016/j.ejor.2017.02.037

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

Naseriparsa, M., Bidgoli, A.-M., & Varaee, T. (2013). A hybrid feature selection method to improve performance of a group of classification algorithms. *International Journal of Computer Applications*, *69*(17), 28–35. https://doi.org/10.5120/12065-8172

Niu, K., Zhang, Z., Liu, Y., & Li, R. (2020). Resampling ensemble model based on data distribution for imbalanced credit risk evaluation in P2P lending. *Information Sciences*, *536*, 120–134. https://doi.org/10.1016/j.ins.2020.05.040

Norat, R., Wu, A. S., & Liu, X. (2023). Genetic algorithms with self-adaptation for predictive classification of Medicare standardized payments for physical therapists. *Expert Systems with Applications*, *218*, Article 119529. https://doi.org/10.1016/j.eswa.2023.119529

Osanaiye, O., Cai, H., Choo, K.-K. R., Dehghantanha, A., Xu, Z., & Dlodlo, M. (2016). Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. *EURASIP Journal on Wireless Communications and Networking*, *2016*, Article 130. https://doi.org/10.1186/s13638-016-0623-3

Ouaderhman, T., Chamlal, H., & Janane, F. Z. (2024). A new filter-based gene selection approach in the DNA microarray domain. *Expert Systems with Applications*, *240*, Article 122504. https://doi.org/10.1016/j.eswa.2023.122504

Palma-Mendoza, R.-J., Rodriguez, D., & de-Marcos, L. (2018). Distributed relieff-based feature selection in Spark. *Knowledge and Information Systems*, *57*(1), 1–20. https://doi.org/10.1007/s10115-017-1145-y

Pashaei, E., & Pashaei, E. (2022). Hybrid binary arithmetic optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical data. *The Journal of Supercomputing*, *78*(13), 15598–15637. https://doi.org/10.1007/s11227-022-04507-2

Qian, H., Wang, B., Yuan, M., Gao, S., & Song, Y. (2022). Financial distress prediction using a corrected feature selection measure and gradient boosted decision tree. *Expert Systems with Applications*, *190*, Article 116202. https://doi.org/10.1016/j.eswa.2021.116202

Rao, C., Liu, M., Goh, M., & Wen, J. (2020). 2-stage modified random forest model for credit risk assessment of P2P network lending to "Three Rurals" borrowers. *Applied Soft Computing*, *95*, Article 106570. https://doi.org/10.1016/j.asoc.2020.106570

Ros, F., Riad, R., & Guillaume, S. (2023). PDBI: A partitioning Davies-Bouldin index for clustering evaluation. *Neurocomputing*, *528*, 178–199. https://doi.org/10.1016/j.neucom.2023.01.043

Sahu, B., & Dash, S. (2024). Optimal feature selection from high-dimensional microarray dataset employing hybrid IG-Jaya model. *Current Materials Science*, *17*(1), 21–43. https://doi.org/10.2174/2666145416666230124143912

Said, R., Elarbi, M., Bechikh, S., Coello Coello, C. A., & Said, L. B. (2023). Discretization-based feature selection as a bilevel optimization problem. *IEEE Transactions on Evolutionary Computation*, *27*(4), 893–907. https://doi.org/10.1109/TEVC.2022.3192113

Sankhwar, S., Gupta, D., Ramya, K. C., Sheeba Rani, S., Shankar, K., & Lakshmanaprabu, S. K. (2020). Improved grey wolf optimization-based feature subset selection with fuzzy neural classifier for financial crisis prediction. *Soft Computing*, *24*(1), 101–110. https://doi.org/10.1007/s00500-019-04323-6

Seijo-Pardo, B., Porto-Díaz, I., Bolón-Canedo, V., & Alonso-Betanzos, A. (2017). Ensemble feature selection: Homogeneous and heterogeneous approaches. *Knowledge-Based Systems*, *118*, 124–139. https://doi.org/10.1016/j.knosys.2016.11.017

Song, Q., Ni, J., & Wang, G. (2013). A fast clustering-based feature subset selection algorithm for high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, *25*(1), 1–14. https://doi.org/10.1109/TKDE.2011.181

Sun, J., Lee, Y.-C., Li, H., & Huang, Q.-H. (2015). Combining B&B-based hybrid feature selection and the imbalance-oriented multiple-classifier ensemble for imbalanced credit risk assessment. *Technological and Economic Development of Economy*, *21*(3), 351–378. https://doi.org/10.3846/20294913.2014.884024

Tran, B., Xue, B., & Zhang, M. (2019). Variable-length particle swarm optimization for feature selection on high-dimensional classification. *IEEE Transactions on Evolutionary Computation*, *23*(3), 473–487. https://doi.org/10.1109/TEVC.2018.2869405

Tsafrir, T., Cohen, A., Nir, E., & Nissim, N. (2023). Efficient feature extraction methodologies for unknown MP4-Malware detection using machine learning algorithms. *Expert Systems with Applications*, *219*, Article 119615. https://doi.org/10.1016/j.eswa.2023.119615

Tsai, C.-F., Sue, K.-L., Hu, Y.-H., & Chiu, A. (2021). Combining feature selection, instance selection, and ensemble classification techniques for improved financial distress prediction. *Journal of Business Research*, *130*, 200–209. https://doi.org/10.1016/j.jbusres.2021.03.018

Tsai, C.-F., Chen, K.-C., & Lin, W.-C. (2024). Feature selection and its combination with data over-sampling for multi-class imbalanced datasets. *Applied Soft Computing*, *153*, Article 111267. https://doi.org/10.1016/j.asoc.2024.111267

Wang, H., & Hong, M. (2015). Distance variance score: An efficient feature selection method in text classification. *Mathematical Problems in Engineering*, *2015*, 1–10. https://doi.org/10.1155/2015/695720

Wang, D., Tan, D., & Liu, L. (2018a). Particle swarm optimization algorithm: An overview. *Soft Computing*, *22*(2), 387–408. https://doi.org/10.1007/s00500-016-2474-6

Wang, D., Zhang, Z., Bai, R., & Mao, Y. (2018b). A hybrid system with filter approach and multiple population genetic algorithm for feature selection in credit scoring. *Journal of Computational and Applied Mathematics*, *329*, 307–321. https://doi.org/10.1016/j.cam.2017.04.036

Xie, Y., Peng, L., Chen, Z., Yang, B., Zhang, H., & Zhang, H. (2019). Generative learning for imbalanced data using the Gaussian mixed model. *Applied Soft Computing*, *79*, 439–451. https://doi.org/10.1016/j.asoc.2019.03.056

Yang, G., Deng, S., Chen, X., Chen, C., Yang, Y., Gong, Z., & Hao, Z. (2023). RESKM: A general framework to accelerate large-scale spectral clustering. *Pattern Recognition*, *137*, Article 109275. https://doi.org/10.1016/j.patcog.2022.109275

Yu, L., Yu, L., & Yu, K. (2021). A high-dimensionality-trait-driven learning paradigm for high dimensional credit classification. *Financial Innovation*, *7*, Article 32. https://doi.org/10.1186/s40854-021-00249-x

Yu, L., Zhang, X., & Yin, H. (2022). An extreme learning machine based virtual sample generation method with feature engineering for credit risk assessment with data scarcity. *Expert Systems with Applications*, *202*, Article 117363. https://doi.org/10.1016/j.eswa.2022.117363

Yun, K. K., Yoon, S. W., & Won, D. (2021). Prediction of stock price direction using a hybrid GA-XGBoost algorithm with a three-stage feature engineering process. *Expert Systems with Applications*, *186*, Article 115716. https://doi.org/10.1016/j.eswa.2021.115716

Zhang, X., Wu, G., Dong, Z., & Crawford, C. (2015). Embedded feature-selection support vector machine for driving pattern recognition. *Journal of the Franklin Institute*, *352*(2), 669–685. https://doi.org/10.1016/j.jfranklin.2014.04.021

Zhang, X., Yu, L., Yin, H., & Lai, K. K. (2022). Integrating data augmentation and hybrid feature selection for small sample credit risk assessment with high dimensionality. *Computers & Operations Research*, *146*, Article 105937. https://doi.org/10.1016/j.cor.2022.105937

Zhang, X., & Yu, L. (2024). Consumer credit risk assessment: A review from the state-of-the-art classification algorithms, data traits, and learning methods. *Expert Systems with Applications*, *237*, Article 121484. https://doi.org/10.1016/j.eswa.2023.121484

Zhao, B., Yang, D., Karimi, H. R., Zhou, B., Feng, S., & Li, G. (2023). Filter-wrapper combined feature selection and adaboost-weighted broad learning system for transformer fault diagnosis under imbalanced samples. *Neurocomputing*, *560*, Article 126803. https://doi.org/10.1016/j.neucom.2023.126803

Zhu, J., Wu, X., Yu, L., & Ji, J. (2024). Improved RBM-based feature extraction for credit risk assessment with high dimensionality. *International Transactions in Operational Research*, (2024), 1–26. https://doi.org/10.1111/itor.13467

Zorarpaci, E. (2024). A fast intrusion detection system based on swift wrapper feature selection and speedy ensemble classifier. *Engineering Applications of Artificial Intelligence*, *133*, Article 108162. https://doi.org/10.1016/j.engappai.2024.108162

APPENDIX

Table A1 presents the complete details of the ablation experiment. It provides a clear overview of each component after ablation, ensuring easy comprehension for the reader.

Table A1. Comparison of ablation experiments

FS methods	Classification Methods	The CUP dataset				The Kaggle dataset			
		ACC	Precision	G-mean	AUC	ACC	Precision	G-mean	AUC
Proposed CBTFS	LDA	0.6282	**0.9215(3)**	0.6830	0.6911	0.4093	0.9167	0.5110	0.5914
	LogR	0.6029	**0.9223(2)**	0.6685	0.6808	0.3322	**0.9420(1)**	0.4174	0.5680
	KNN	0.6136	0.9106	0.6649	0.6720	0.4036	0.8201	0.4815	0.5150
	DT	0.6777	0.8847	0.6594	0.6602	0.5782	0.8356	0.5525	0.5541
	RF	0.7051	**0.9246(1)**	**0.7282(1)**	**0.7294(1)**	0.5692	0.8575	0.5854	0.5862
	XGBoost	**0.7186(2)**	0.9136	**0.7192(3)**	**0.7195(3)**	**0.6803(3)**	0.8557	**0.5969(3)**	**0.6085(2)**
	AdaBoost	**0.7186(2)**	0.9148	**0.7209(2)**	**0.7213(2)**	0.6190	0.8558	0.5935	0.5949
	Bagging	**0.7153(3)**	0.9022	0.7004	0.7010	0.6224	0.8270	0.5282	0.5437
	Average	0.6725	**0.9117**	**0.6930**	**0.6969**	0.5267	0.8638	0.5333	0.5702
w/o AdaBoost, RF, and XGBoost	LDA	0.2684	0.8363	0.3149	0.5069	0.5922	0.7906	0.4040	0.4883
	LogR	0.5200	0.8408	0.5376	0.5541	**0.7550(2)**	0.8080	0.3079	0.5188
	KNN	0.5940	0.8277	0.5372	0.5459	0.2165	0.8202	0.1613	0.5010
	DT	0.5604	0.7991	0.4362	0.4792	0.6245	0.7985	0.3274	0.4962
	RF	0.4246	0.7746	0.4495	0.4608	**0.7874(1)**	0.7999	0.1113	0.4973
	XGBoost	0.2143	0.6381	0.0778	0.4973	0.3506	0.8754	0.3638	0.5316
	AdaBoost	0.2036	0.7030	0.0772	0.5018	0.5425	0.8284	0.4444	0.5250
	Bagging	0.5514	0.7719	0.3894	0.4405	0.6529	0.7968	0.3636	0.4953
	Average	0.4170	0.7739	0.3524	0.4983	0.5652	0.8147	0.3104	0.5066
w/o IMST, RF, and XGBoost	LDA	0.5627	0.9140	0.6361	0.6539	0.4108	**0.9289(3)**	0.5131	0.5990
	LogR	0.5963	0.9191	0.6613	0.6737	0.4159	0.9270	0.5183	0.6002
	KNN	0.6103	0.8995	0.6516	0.6562	0.3784	0.8794	0.4737	0.5569
	DT	0.6635	0.8838	0.6521	0.6528	0.5320	0.8526	0.5680	0.5722
	RF	0.6952	0.9135	0.7081	0.7087	0.5105	0.8853	0.5843	0.6045
	XGBoost	0.7147	0.9045	0.7024	0.7031	0.5819	0.8800	**0.6174(1)**	0.6012
	AdaBoost	0.6767	0.9162	0.7029	0.7047	0.4870	0.8853	0.5685	0.5966
	Bagging	0.6969	0.8993	0.6878	0.6883	0.5513	0.8671	0.5907	0.5955
	Average	0.6520	0.9062	0.6752	0.6801	0.4834	0.8882	0.5542	**0.5907**
w/o IMST, AdaBoost, and XGBoost	LDA	0.5754	0.9020	0.6352	0.6459	0.3685	**0.9299(2)**	0.4649	0.5802
	LogR	0.5911	0.9111	0.6518	0.6627	0.3644	0.9290	0.4598	0.5778
	KNN	0.6357	0.8951	0.6581	0.6597	0.3226	0.9017	0.4048	0.5490
	DT	0.6783	0.8794	0.6484	0.6505	0.4981	0.8596	0.5593	0.5729
	RF	0.6814	0.9098	0.6970	0.6979	0.4527	0.8998	0.5480	0.5971
	XGBoost	0.7072	0.8979	0.6889	0.6900	0.5063	0.8862	0.5821	0.6040
	AdaBoost	0.6782	0.9117	0.6981	0.6992	0.4391	0.9008	0.5365	0.5927
	Bagging	0.6936	0.8937	0.6774	0.6784	0.4985	0.8782	0.5724	0.5933
	Average	0.6551	0.9000	0.6693	0.6730	0.4312	**0.8981**	0.5159	0.5833
w/o IMST, AdaBoost, and RF	LDA	0.6597	0.9072	0.6840	0.6856	0.4229	0.9211	0.5251	0.6002
	LogR	0.6439	0.9110	0.6805	0.6841	0.4324	0.9212	0.5346	0.6044
	KNN	0.6556	0.8962	0.6679	0.6685	0.4227	0.8683	0.5135	0.5621
	DT	0.6983	0.8823	0.6581	0.6615	0.5527	0.8414	0.5598	0.5603
	RF	0.7088	0.9047	0.7009	0.7012	0.5441	0.8778	0.5927	**0.6068(3)**
	XGBoost	0.7237(1)	0.8971	0.6927	0.6948	0.6169	0.8656	**0.6101(2)**	**0.6108(1)**
	AdaBoost	0.6980	0.9143	0.7103	0.7110	0.5165	0.8854	0.5879	0.6065
	Bagging	0.7144	0.8943	0.6850	0.6869	0.5857	0.8592	0.5930	0.5935
	Average	**0.6878**	0.9008	0.6849	0.6867	0.5117	0.8800	**0.5652**	0.5167