Expert Systems With Applications 195 (2022) 116624

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A two-stage hybrid credit risk prediction model based on XGBoost and graph-based deep neural network

Jiaming Liu [a,*], Sicheng Zhang [b], Haoyue Fan [b]

[a] School of International Economics and Management, Beijing Technology and Business University, Beijing 100048, China
[b] School of Economics and Management, Beijing University of Chemical Technology, Beijing 100029, China

ARTICLE INFO

Keywords:
Feature engineering
Graph-based deep neural network
Hybrid model
XGBoost
Credit risk prediction

ABSTRACT

The credit risk prediction technique is an indispensable financial tool for measuring the default probability of credit applicants. With the rapid development of machine learning and the application of big data, increasingly sophisticated models have been designed to construct effective credit risk prediction models. In this study, we propose a two-stage hybrid model to enhance the prediction performance of credit risk. First, to make full use of the classified information hidden in credit data, we employ XGBoost to linearize and transform the original features into a high-dimensional sparse feature matrix. Second, to effectively process the transformed high-dimensional data and to discover the relationships between the features, a recently proposed graph-based neural network (forgeNet) model, which is good at addressing high-dimensional data, is deployed to predict the credit risk. The real-world credit data of the Lending Club for the period from 2007 to 2016 were collected and partitioned based on the economic cycle to validate the robustness of the proposed model. The experimental results show that feature transformation and feature graph mining are two pragmatic processes for credit risk prediction when analyzing credit data. Furthermore, the proposed model is robust against different economic cycles and achieves the best average prediction results of 87.52%, 93.13% and 85.59% in terms of accuracy, F1-score, and G-mean compared with other benchmarks, including individuals, hybrid models, and ensembles. The average performance of the proposed model rose by 6.14, 7.59 and 6.18 percentage points, respectively, which demonstrates the outperformance of the proposed two-stage model in credit risk prediction applications

1. Introduction

Online financial innovation services and traditional financial services play a major role in serving the real economy, and both services have their own natural advantages in the financial industry. As a product of financial innovation services, the online peer-to-peer (P2P) lending platform has rapidly spread to the Americas, Asia, and other regions since 2005. After more than ten years of development, the business mode of P2P has gradually stabilized around the world, and the supporting regulatory systems of various countries have become increasingly mature.

P2P mainly lends money to applicants by borrowing demands through electronic trading platforms. Because P2P lending platforms are characterized by mortgage-free personal loans, investors will bear enormous losses when the lender defaults. The credit risk prediction technique is an important tool to estimate the default probability of an applicant according to the personal and demographic information on the applicants. Financial institutions mainly judge the loan qualifications of the applicants through expert experience from previous decisions. However, with the rapid development of big data and artificial intelligence, historical data from applicants are employed to learn an advanced classification model for credit risk prediction tasks based on the assumption that the credit status of a borrower is portrayed by his or her historical behavior.

The credit risk prediction task is a typical binary classification problem that divides the population of applicants into two groups: a 'good' class and a 'bad' class. There are a myriad of studies that investigate dichotomous prediction models to predict the default probability of the applicants. Statistical methods and machine learning are two commonly used techniques, such as logistic regression (Dumitrescu et al., 2021), support vector machines (Luo et al., 2020; Yu et al., 2020), decision trees (Golbayani et al., 2020; Xia et al., 2020), and neural networks (Shen et al., 2019; Liang and Cai, 2020). Even the recently popular deep neural network (DNN) has also been applied in this field

* Corresponding author.
E-mail addresses: liujiaming@btbu.edu.cn (J. Liu), sczhang@mail.buct.edu.cn (S. Zhang).

https://doi.org/10.1016/j.eswa.2022.116624
Received 15 April 2021; Received in revised form 28 August 2021; Accepted 27 January 2022
Available online 1 February 2022

(Shen et al., 2020; Fu et al., 2020).

However, to weaken the complex nonlinear relationship between the features and to make full use of the classification information contained in the feature set, feature engineering is an essential task before constructing the credit risk prediction model (Zhang et al., 2019). Feature engineering performs a series of engineering processes on the original data and refines them into features, which are used as inputs for algorithms and models (Nargesian et al., 2017). It mainly includes feature selection, feature extraction and feature transformation. Feature selection enhances the prediction performance of the model by discarding irrelevant or redundant features to find a subset of features that have strong discriminative ability (Tsai et al., 2021). Filter, wrapper and embedded are three categories of feature selection methods according to different selection processes (Kozodoi et al., 2019). Feature extraction extracts effective classification information from the original features by analyzing the orthogonal or manifold properties of the features to build a new feature set for model construction (Orsenigo and Vercellis, 2013). The most representative methods are principal component analysis (PCA) and manifold learning (Wang and Wu, 2017; Dastile et al., 2020).

Although these feature engineering methods have achieved relatively good results in credit risk prediction applications, these methods still have several problems. First, in general, there is a complex intrinsic nonlinear relationship between credit features and credit risk (Huang and Kou, 2014). Most of the feature selection methods and feature extraction methods cannot linearize the original features to better serve the prediction model. Second, these methods usually assume that the features are independent of one another, thus ignoring the dependencies between features, and such relationship information is also a key factor in building classification models (Fu et al., 2021).

Because of the above problems, more sophisticated feature engineering methods should be proposed instead of using only a single feature processing method. Therefore, this study proposed a two-stage hybrid credit risk prediction model to address feature engineering. First, the tree-based model is good at capturing the nonlinear relationship between predictors and targets, which has been proven to have a good feature linearization ability (He et al., 2014). Therefore, we employ a well-known class of ensemble tree-based models, XGBoost, to conduct the first stage of the feature linearization process. Then, with the forgeNet model proposed by Kong and Yu, 2020, to fully mine the dependency relationships between features, we construct a feature graph by investigating the tree split process among the features with XGBoost. A node represents a feature, and edges represent directed relationships between features. Finally, we embed the feature graph into a deep neural network to build the credit risk prediction model. The experimental results from the real-world data show that feature transformation and feature graph mining are two pragmatic processes for credit risk prediction. Our proposed two-stage hybrid prediction model is competent for the task of credit risk prediction.

The main contributions of this study are highlighted as follows: First, to fully mine the classification knowledge contained in credit data, we transform the intricate complex nonlinear features into linear features by using XGBoost. Second, to incorporate the complex network structure of the transformed features into a DNN, we construct the feature graph by reorganizing the transformed feature space and applying forgeNet for credit risk prediction tasks by combining the feature graph and DNN. Third, we deploy the graph-based DNN and propose a new two-stage hybrid model for credit risk prediction.

The goals of this study are three-fold. **Goal 1**: To construct a discriminant feature set and develop a new feature engineering method by fusing feature transformation and feature graph mining. **Goal 2**: To build an effective credit risk prediction model and propose a two-stage hybrid credit risk prediction model by using a deep multilayer neural network. **Goal 3**: To verify the prediction performance of the proposed model and to compare the prediction results from the real-world credit data.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature on credit risk prediction models and preliminaries. Section 3 introduces the proposed two-stage credit risk prediction model. Section 4 describes the experimental credit data and experimental environment. Section 5 discusses the experimental results, and Section 6 provides the conclusions.

2. Literature review

2.1. Feature engineering methods in credit risk prediction models

In this section, we review some literature related to this study. Credit data are usually characterized by their high dimensionality and complex nonlinearity. To diminish these negative effects, feature selection/engineering are commonly used in credit scoring systems. Filter, wrapper and embedded are three types of feature selection methods (Haq et al., 2021, Gokalp et al., 2020, Jiménez-Cordero et al., 2021).

The filter method scores each dimension of the features based on certain criteria, such as the mutual information (Kim, 2016) and the correlation coefficient (Mokhtia et al., 2021). In other words, it assigns a weight to each dimension of the features, which represents the importance of the features, and then sorts them according to their weights. The filter method selects features according to the weight first and then trains the learner; as a result, the process of feature selection by using a filter is independent of the learner (Haq et al., 2021). Some common filter methods include the chi-squared test (BinSaeedan and Alramlawi, 2021), information gain (Jadhav et al., 2018), Boruta (Kursa and Rudnicki, 2010), and correlation coefficient scores (Jiang et al., 2020). Because the weight assigned to a feature according to certain criteria does not truly represent the importance of the feature in the model, the filter method usually does not perform well in the experiments.

The wrapper method treats the selection of feature subsets as a search optimization problem to generate different combinations and then evaluates the combinations and compares them with other combinations. The selection of subsets is regarded as an optimization problem, which can be solved by many optimization algorithms, especially some heuristic optimization algorithms, such as genetic algorithms (Lappas and Yannacopoulos, 2021), particle swarm optimization (Paul et al., 2021), greedy algorithms (Jiang, 2018), and differential evolution (Tarkhaneh et al., 2021). The characteristic of the wrapper is that the learner is directly used as the evaluation function for the feature selection, and the optimal feature subset is selected for a specific learner. Recursive feature elimination is one of the most representative wrapper methods (Rtayli and Enneya, 2020).

The embedded method learns the best features to improve the accuracy of the model when the model is given (Jiménez-Cordero et al., 2021). In other words, in the process of determining the model, those features that are important for the training of the model are selected. The L1-based feature selection method is a type of embedded method that reduces the number of features by imposing the L1-norm penalty on the linear model (Cui et al., 2021b). There are also L2-based feature selection and elastic-net methods, depending on the setting of the penalty term for the linear model (Cui et al., 2021a). Tree-based feature selection is another prevalent embedded method that computes the impurity of the tree leaves iteratively, which can be used to discard irrelevant features (Zhou et al., 2021). However, the embedded method requires specific learners and provides less feasibility in the use of the method.

In addition to the feature selection method, feature extraction is another feature engineering strategy that reorganizes and extracts effective classification information from the original features to build a learner. Principal component analysis (PCA) and locally linear embedding (LLD) are two prevailing feature extraction techniques (Orsenigo and Vercellis, 2013). PCA aims at extracting the most meaningful components by preserving the most variance in the data. The main drawback of PCA is that it is a type of statistical technique for linear dimensional reduction. LLD is a type of manifold learning for nonlinear

Table 1
The advantages and disadvantages of typical feature engineering methods to address features.

Method	Reference	Advantages	Disadvantages
Filter method	[13], [34]	Simple and easy to implement	Irrelevant to the learner and can result in no performance improvement or even degrade the performance
Wrapper method	[11], [16], [41]	Uncovers the feature subset through the learner; the performance of the feature subset is usually better	Computationally more expensive than the filter methods
Embedded method	[4], [5], [19]	Conducts the feature selection process automatically when building the model	Only applicable for specific algorithms and models
PCA	[34]	Reduce the computational cost and make the reduction results easier to understand	Performs with poor performance when the features are nonlinearly correlated
LLD	[34], [47]	Proxy to sparse matrix eigen-decomposition, less computational complexity, easy to implement	Sensitive to the choice of the number of nearest neighbor samples
Tree-based model	[48]	Less computational complexity, easy to understand	Perform poorly when addressing data that has relatively strong feature correlations

dimensional reduction. LLD attempts to find a low-dimensional manifold to handle complex nonlinear structures of the data (Zhang et al., 2018). An increasing number of novel feature engineering strategies are used in credit risk evaluation through feature generation, feature transformation, feature graph mining and so on (Lucas et al. (2020), Zhang et al. (2019)).

The advantages and disadvantages of the relevant techniques are summarized in Table 1.

2.2. eXtreme gradient boosting

Chen and Guestrin (2016) proposed eXtreme gradient boosting (XGBoost) by modifying gradient boosting trees, which was proposed by Friedman (2001). XGBoost is a family member of boosting ensemble models based on tree learners and has been extensively used in both classification and regression applications (Chang et al., 2018; Xia et al., 2020). In this study, we employ XGBoost to conduct feature transformation and feature graph construction tasks instead of classification or regression issues. XGBoost defines the loss function to minimize the prediction error of the model and designs the regularization function to control the complexity of the model. The theory of XGBoost is given below:

$$Obj^t = L^t + \Omega^t \tag{1}$$

where O_t denotes the objective function of the model, L_t is used to calculate the loss value of the model, and Ω_t denotes the regular function of the model to prevent overfitting.

XGBoost uses an additive ensemble strategy to complement the optimization of the model, and one tree is added into the current XGBoost model in the t-th round of the ensemble. Then, the prediction result from the current model can be written as

$$\hat{y}_i^t = \hat{y}_i^{t-1} + f_t(x_i) \tag{2}$$

where $f_t(x_i)$ represents the function of the tree to be integrated for the current model, and x_i represents the feature space of the samples. Here, $\hat{y}_i^{t-1}$ indicates the current prediction result, and $\hat{y}_i^t$ presents the prediction result for the new model. The objective function can be rewritten as

$$Obj^t = \sum_{i=1}^{N} l\left(y_i, \hat{y}_i^{t-1} + f_t(x_i)\right) + \Omega(f_t) + constant \tag{3}$$

The objective function (3) can be expanded by Taylor expansion. XGBoost uses the second derivative to design the objective function, and then, a new form of objective function is derived as

$$Obj^t = \sum_{i=1}^{N}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t) \tag{4}$$

where $g_i = \partial_{\hat{y}_i^{t-1}} l(y_i, \hat{y}_i^{t-1})$, $h_i = \partial^2_{\hat{y}_i^{t-1}} l(y_i, \hat{y}_i^{t-1})$.

For regular terms, the regularization function is defined by the leaf number T and the vector of scores on the leaves ω_j. The regularization function can be rewritten as

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T}\omega_j^2 \tag{5}$$

After designing the complexity of the trees, the objective function can be rewritten in the form of grouping by each leaf:

$$Obj^t = \sum_{i=1}^{N}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T}\omega_j^2 = \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i\right)\omega_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right)\omega_j^2\right] + \gamma T \tag{6}$$

where I_j is the set of data points assigned to the j-th leaf. By setting $G_j = \sum_{i\in I_j} g_i$ and $H_j = \sum_{i\in I_j} h_i$, Eq. (6) can be further simplified as

$$Obj^t = \sum_{j=1}^{T}\left[G_j\omega_j + \frac{1}{2}\left(H_j + \lambda\right)\omega_j^2\right] + \gamma T \tag{7}$$

Then, we can derive the optimal ω_j^* and the best objective reduction Obj^* as $\omega_j^* = -\frac{G_j}{H_j+\lambda}$,

$$Obj^* = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j + \lambda} + \gamma T \tag{8}$$

Although XGBoost is a mature and applicable model in many application scenarios, some variants, including LightGBM (Ke et al., 2017) and CatBoost (Prokhorenkova et al., 2018), have also been developed with the aim of cost savings and performance improvement. This type of tree ensemble model is also applied in credit risk prediction, and relevant studies can be found in Ma et al., (2018), Sigrist and Hirnschall, (2019), and Chang et al., (2018).

2.3. Deep neural networks

Neural networks have demonstrated powerful classification and recognition capabilities, and feedforward neural networks (FNNs) are one of the basic networks with straightforward structures in the network family (Moghanian et al., 2020). Graph embedding is a type of graph representation learning technique that brings new opportunities to the

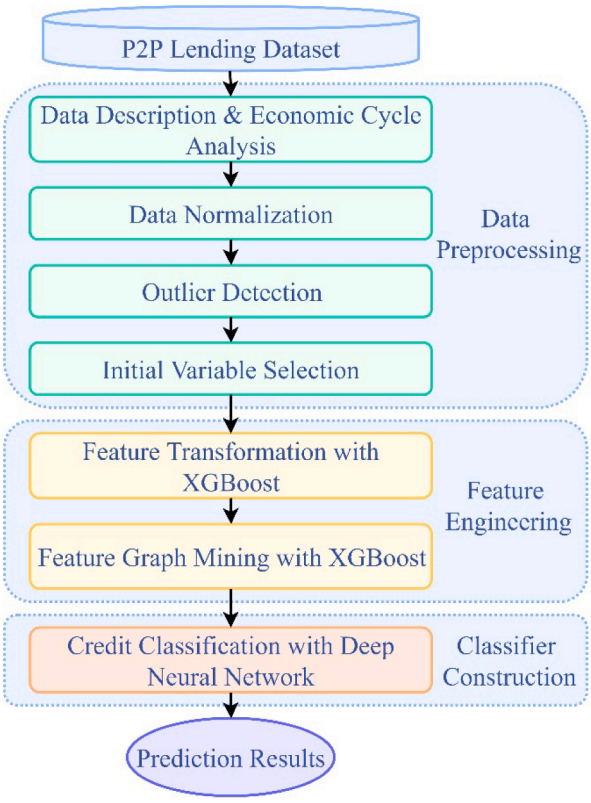


Fig. 1. Framework of the proposed two-stage hybrid model.

optimization of FNN structures. We provide a brief description for graphs and FNNs.

Let $G = (V, E)$ denote the graph, where E is the edge set of G, and V is the vertex set of G. For feature representation learning, the feature space can be expressed as a graph network by mining the relations among features. In the graph network, the features are presented as vertices, and two features are connected if there exists a relation between two features.

The FNN is characterized by its feasible structure and consists of multilayer neurons in general. Let the hidden layer l consist of M neurons, let the hidden layer $l+1$ consist of N neurons, and let the output of the $(l+1)$-*th* hidden layer be expressed as

$$z_n^{l+1} = g_n^{l+1}\left(\sum\nolimits_{m-1}^{M} w_{nm}^{l+1} \bullet \mathrm{x}_{\mathrm{m}}^{\mathrm{l}} + \mathrm{b}_{\mathrm{n}}^{\mathrm{l}+1}\right) \tag{9}$$

where w is the weight between two adjacent layers, b is the bias of the layer, and $g(\bullet)$ is the activation function, which is usually set as a nonlinear differential function such as the sigmoid, ReLU or tanh. The output $\hat{y}$ of the output layer is converted by a $softmax(\bullet)$ function, which can be expressed as

$$\hat{y} = P(y|x) = softmax(z) = \frac{exp(z_i)}{\sum_j exp(z_j)} \tag{10}$$

For graph-based feedforward neural networks, the input layer is modified by a feature graph network, and as a result, the structure of the neural network is modified (Kong and Yu, 2020). Specifically, the output of the first hidden layer is presented as

$$z^1 = g^1\left(w^1 \bullet \mathrm{A} \bullet \mathrm{x} + \mathrm{b}^1\right) \tag{11}$$

where A is the adjacent matrix, which is derived from the feature graph network. In general, the number of layers and the corresponding number of neurons are set empirically. The weights and biases are commonly determined by the stochastic gradient descent method. Most recently, deep learning has reshaped credit risk evaluation through its powerful predictive ability; relevant studies can be found in Shen et al. (2020b), Kim et al. (2019), and Fu et al. (2020).

2.4. Research hypotheses

The objectives of this study are threefold: the first objective is to verify the effectiveness of the transformed features in improving the performance of credit risk prediction, the second objective is to examine the prediction performance of the graph DNN, and the third objective is to validate the superiority of the proposed two-stage hybrid mode. Therefore, we propose three hypotheses that must be verified:

H1: The performance of the DNN when using the transformed features is significantly better than that of the DNN when using the original features. To examine H1, the original dataset is first applied to train and test the performance of the DNN, and then, the new dataset with the transformed features is employed to train and test the performance of the DNN.

H2: The performance of the DNN by incorporating a feature graph is significantly better than that of the DNN by using the transformed features. To examine H2, XGBoost is first applied to construct the feature graph by using the transformed features, and then, the feature graphs are embedded into the DNN to train the graph-based DNN, and the testing data are applied to test the prediction performance of the graph-based DNN.

H3: The performance of the proposed hybrid model is significantly better than that of other benchmarks. To examine H3, the proposed hybrid model, XGB-forgeNet, is trained and validated through feature linearization and feature graph construction processes, and the other benchmarks are built using the original dataset.

3. The proposed model

3.1. Framework of the proposed two-stage hybrid model

To describe the proposed two-stage hybrid model in detail, Fig. 1 provides the overview framework of the proposed hybrid model. The proposed two-stage hybrid model is based on the following components. First, we propose a new feature engineering strategy to handle credit data with a complex intrinsic nonlinear structure. The relationship between the predictors (applicants' personal or demographic information) and credit status (default or nondefault) is often nonlinear (Huang and Kou, 2014). Hence, the construction of a classification model by using the original credit dataset cannot predict credit default accurately. An arbitrary application of the nonlinear classifier will also dismiss important knowledge hidden in the original feature space.

In the feature engineering stage, to build a more separable feature set, we reconstruct the original feature space by using a tree-based ensemble model, which aims to transform the original nonlinear feature set into a linearly separable feature set. Specifically, to substantially reduce the negative effects caused by the nonlinear relationships of the features, XGBoost is used to encode the features and to generate a new feature set based on the leaf distribution of the ensemble tree model. Then, for the second part of the feature engineering stage, because the transformed feature space is highly sparse, the core objective of this part is to diminish the high sparsity structure and to investigate the relationships of the transformed feature set. Specifically, XGBoost is applied to discover this type of relationship and to construct a feature graph. Finally, the graph-based deep neural network model is built based on the feature graph with the aim of incorporating relational knowledge for the DNN model.

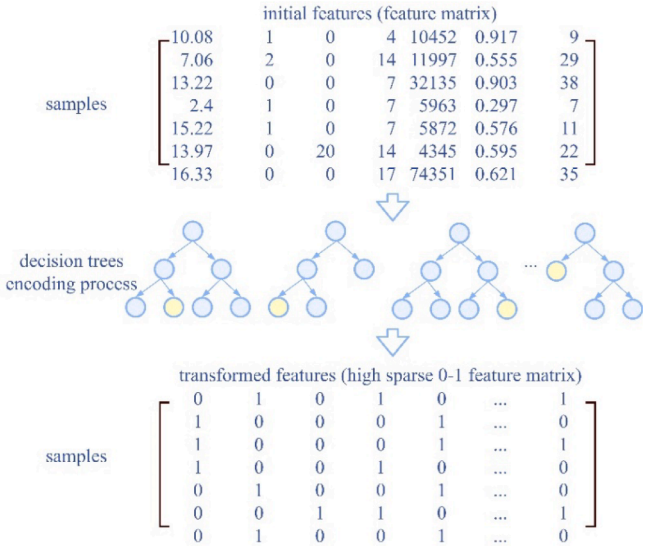


Fig. 2. A vivid depiction of the feature transformation process.

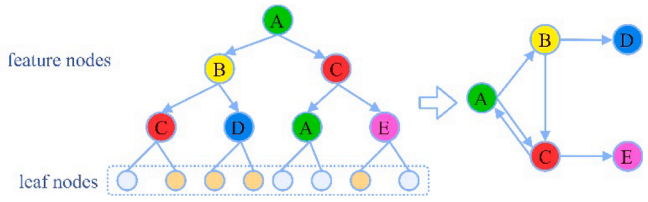


Fig. 3. Feature relation mining by using a tree structure.

3.2. Feature engineering strategy

3.2.1. Feature transformation module

Tree-like structure models use a divide-and-conquer strategy to select the current most representative feature as the split feature. Since the entropy can measure the classification information in a feature, the information gain and the Gini index are often used as splitting strategies in tree models. The classification verdict of the instances is determined according to the proportion of classes on the leaf node where the instance falls. As a type of tree-based ensemble model, XGBoost constructs the base tree iteratively based on the residual of the last round of the ensemble. The instance is encoded as a binary vector through the nodes' distribution of each base tree. Then, the original feature space is transformed as a sparse matrix, where the number of rows corresponds to the number of instances and the number of columns corresponds to the number of leaves of XGBoost.

Assuming that the number of trees is T in XGBoost, the corresponding leaves of each tree are represented as $L_i = \{n_1, n_2, \cdots, n_m\}, i = 1, \cdots, T$, where n_m indicates the number of leaves of the m-th tree. To describe the transformation process of the original features clearly, we take a simple example. Assuming that the number of leaves of the first tree is five and that the sample falls in the third leaf of the tree, we then have $n_1 = \{0,0,1,0,0\}$, which indicates that the sample is transformed to the vector n_1. For a clear illustration, the transformation process is depicted in Fig. 2. The transformation process is intuitive and effective by mapping the original feature space into a new feature space through a tree structure. The derived high-dimensional sparse feature set is more separable than the original feature set, which is credited to the property of the ensemble tree structure.

3.2.2. Feature graph mining module

The credit data are transformed into a sparse high-dimensional feature set after the feature transformation process. To discover the relationships among the transformed features, XGBoost is used as the relation mining tool to transform the tree structure into a directed feature graph, where the feature presents the node in the feature graph and the tree branch indicates that there exists a relation between the parent node and child node. For a clear illustration, Fig. 3 presents an illustration example to depict the feature relationship discovered by the tree structure model.

Assuming that there are four features (A, B, C, D) for a tree, the feature graph transformation process is depicted in Fig. 3. According to the structure of the base trees, the trees are extracted as a feature graph with directions, where the edges indicate the split branch of each node

Table 2
The pseudocode of the forgeNet model.

```
Input: a set of samples: S
  Tree-based ensemble model: XGBoost
  Deep learning model: DNN
  Number of learning epochs: EP
  Number of layers: L
Output: The default probability of the applicant.
Process:
  XGBoost ← build xgboost with S
  forest ← reorganize xgboost into forest
  selected_features ← forest_to_list (forest)
  initialize rows, biases, weights, layers, learning rate
  train_data ← data[rows, selected_features]
  feature_graph ← zeros([train_data[:rows,:], train_data[:rows,:]])
  feature_graph ← feature_graph[selected,:][:, selected]
  for layer ← 1 to L do
    layers ← add_layer (weights, feature_graph, biases)
  end_for
  for epoch ← 1 to EP do
    x_tmp, y_tmp ← shuffle (x_train_data, y_train_data)
    DNN ← run (x_tmp, y_tmp, layers) % construct DNN
  end_for
```

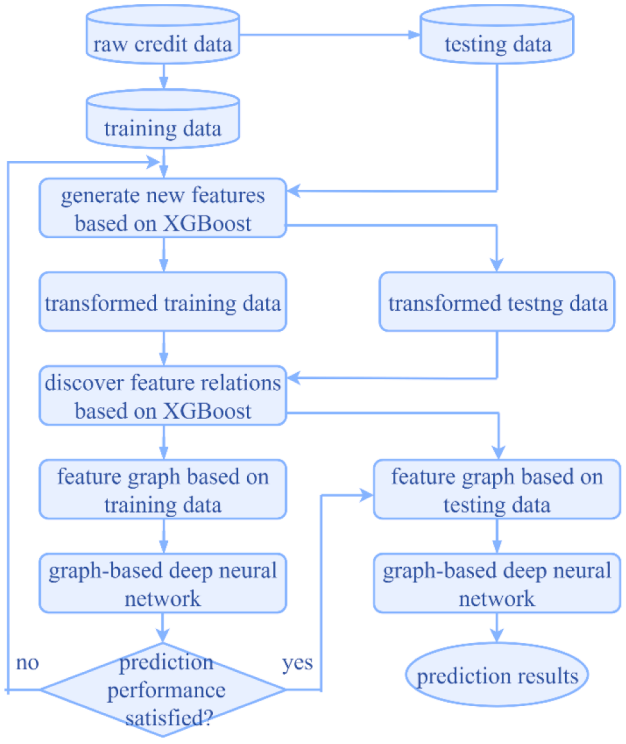


Fig. 4. Flowchart of the proposed model.

Table 3
Pseudocode of the proposed two-stage credit risk prediction model.

```
Input: a set of training samples: S
    Tree-based ensemble model: XGBoost
    Deep learning model: DNN
    Number of learning epochs: EP
    Number of batch size: BS
    Number of layers: L
Output: The default probability of the applicant.
Process:
    xtrain, ytrain, xtest, ytest ← preprocess_split (S)
    XGBoost ← fit (xtrain, ytrain)
    train_leaves, test_leaves ← XGBoost.apply (xtrain, ytrain)
    train_data ← XGBoost.transform (train_leaves, test_leaves) % feature linearization
    initialize weights, bias, layer_size, learning_rate, batch_size, epochs
    XGBoost ← fit (train_data)
    feature_graph ← build feature graph based on XGBoost % feature graph discovery
    for layer ← 1 to L do
      layers ← add_layer (weights, feature_graph, biases)
    end_for
    for epoch ← 1 to EP do
      x_tmp, y_tmp ← shuffle (x_train_data, y_train_data)
      for batch ← 1 to BS do:
        batch_x, batch_y ← partition (x_tmp, y_tmp)
        DNN ← run (batch_x, batch_y, layers) % construct DNN
      end_for
    y_pred ← DNN.predict (xtest, ytest)
    end_for
```

to other nodes and the nodes present the features. The high-dimensional sparse features are reconstructed as a feature graph for further use.

3.3. The forgeNet model

Kong and Yu (2020) proposed forgeNet to handle biometric information and obtained excellent performance in disease diagnosis. Therefore, forgeNet is used to investigate the complex relations among sparse variables and to construct a deep neural network classification model by using a feature graph. ForgeNet mainly consists of two steps. The first step is feature graph construction by using a tree-based ensemble model to mine the feature relations. In the second stage, the feature graph derived from the first stage is embedded into the deep forward neural network. In other words, instead of using a full connection of the input layer of the DNN, a sparse connection with feature graph information is introduced as the input layer. The graph-based DNN is then built according to the extracted feature graph. The pseudocode of the forgeNet model is presented as follows: Table 2.

3.4. Process and algorithm of the model

Because the features of raw credit data are not all linear, the original features must be transformed through a feature linearization method to generate a new feature set. XGBoost is deployed to transform the raw features by using the training data, while testing data are also transformed through the trained XGBoost. Then, the transformed training and testing data are derived from the feature linearization process. Because the transformed data have high sparsity and the features are not independent from each other, they must be converted to a feature graph to discover the relations between the features. Therefore, a forest structure model is utilized to investigate the relationships using the XGBoost model. The edges of the feature graph are the split direction of the tree, and the nodes of the feature graph are the features of the tree. The transformed features are then converted into a feature graph that contains relational information. Finally, the deep neural network is modified by the feature graph and constructed to predict the credit risk of the applicants. Fig. 4 depicts the flowchart of the proposed model. The pseudocode of the proposed two-stage credit risk prediction model is shown in Table 3.

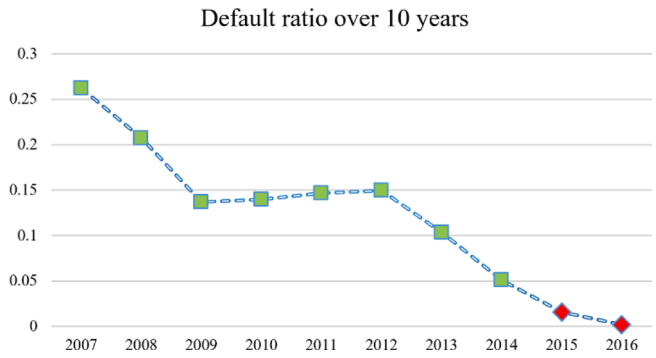


Fig. 5. The tendency of the default ratio over 10 years.

Table 4
Basic description of the datasets over two periods.

Dataset period	Majority samples	Minority samples	Variables	Imbalanced ratio
2007–2014	426,062	40,225	20–1	10:1
2015–2016	576,208	6,656	53–1	close to 90:1

4. Data preparation

This section describes the dataset used in this study, including dataset introduction, data preprocessing and data description.

4.1. Dataset

Credit data are collected from Lending Club[1], which is a popular online P2P platform in the US. The platform has kept the transaction data and the applicants' information since 2007. The platform stores a very large amount of credit data, which provides sufficient data for the establishment of a credit risk prediction model. We collect credit data from 2007 to 2016 over 10 years. Fig. 5 plots the trend of the default ratio over 10 years. Fig. 5 shows that the default ratio was over 5% between 2007 and 2014, while the default ratio was relatively small during 2015 and 2016. To validate the prediction performance of the proposed model and examine the robustness of the proposed model, we divide the dataset into two periods: 2007–2014 and 2015–2016. The first period experienced an economic recession, while the economic environment of the second period was relatively stable.

4.2. Data preprocessing

The collected data mainly face three issues: missing value imputation, outlier detection and variable selection. Data preprocessing is an essential step in addressing these issues to establish a productive credit risk prediction model. First, the nominal variable is encoded with numerical values based on a one-hot encoding approach. Second, the variables with missing proportions over 60% are removed; otherwise, the missing values are imputed with the mean value or mode value for numerical variables and nominal variables, respectively. Third, the interquartile range (IQR) method is applied to detect outliers. The selection rule is defined below:

$$IQR = Q3 - Q1$$

$$Outliers = value < (Q1 - 1.5 \times IQR) or value > (Q3 + 1.5 \times IQR)$$

[1] https://www.lendingclub.com/info/download-data

Table 5
Confusion matrix.

		Actual category	
		Nondefault	Default
Prediction category	Nondefault	TP	FP
	Default	FN	TN

where Q1 and Q3 are the lower quartile and upper quartile of the variable, respectively, and IQR is defined as the interquartile range. The values located outside of the range are deemed to be outliers. Then, the outliers are replaced by the lower quartile and upper quartile.

Fourth, the variables are filtered beforehand according to the correlation coefficients among the variables. If the correlation coefficient between two variables was over 0.8, then one variable was removed. After the data cleaning process, there were 466,287 and 582,864 samples in the dataset corresponding to two different periods. The basic information on the majority group and minority group is listed in Table 4.

From the imbalanced ratio, it can be observed that the ratio between majority samples and minority samples is highly skewed. To balance the dataset, we used a random undersampling approach to select the samples from the majority class. The ratio between majority and minority was set to 1:2 for further experiments.

To test the effectiveness of the proposed model, we use 80% of the dataset to train the classification models and leave 20% of the dataset to examine the performance of each model. Ten rounds of holdout experiments are conducted to overcome the prediction result bias.

5. Experiments

This section describes the experiments, including benchmarks and configuration, numerical comparison, and discussions.

5.1. Benchmarks and configurations

5.1.1. Benchmarks

To validate the effectiveness of the proposed two-stage hybrid model in the credit risk prediction task, we compare our proposed model with six state-of-the-art classification models and a dataset from two different periods. In terms of classification models, logistic regression (LR), support vector machines (SVMs) and deep forward neural networks (DNNs) are commonly employed as individual benchmarks in credit risk prediction. The prediction performance of LR and DNN are also tested by using the transformed features, and the hybrid classifiers are denoted as XGB + LR and XGB + DNN, respectively. We also employ random forest (RF) and extreme gradient boosting machine (XGBoost) as ensemble benchmarks. In terms of the dataset, the datasets from two periods are collected and deployed to test the superiority of the proposed model.

All of the experiments are run on a laptop running Windows 10 with 2.50 GHz Intel Core i5 CPU, 8 GB of RAM. The prevalent python machine learning toolbox scikit-learn[2] is used for model deployment, including LR, SVM, and RF. The XGBoost model comes from the xgboost package[3]. DNN and forgeNet are constructed based on the TensorFlow framework[4] and Kong[5].

5.1.2. Hyperparameter setting

To make a fair comparison, we use the same parameter tuning strategy - grid search method - to find the optimal hyperparameters of the models. For the SVM with the RBF kernel, the search range of the RBF kernel parameter γ varied between $\gamma \in \{2^{-6}, 2^{-5}, \cdots, 2\}$, and the cost parameter c varied in the range $c \in \{2^{-3}, 2^{-2}, \cdots, 2^{3}\}$. For RF, the number of trees for integration plays an important role in model construction.

Table 6
Ten rounds of holdout prediction results in terms of accuracy (%) for 2007–2014 (Period 1).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	74.58	76.27	79.62	83.03	87.51	85.3	87.57	**88.25**
2	74.56	77.09	80.47	84.01	87.7	85.88	88.33	**89.33**
3	74.68	76.47	79.72	83.85	88.33	86.01	88.03	**88.76**
4	74.3	76.09	79.67	83.46	87.69	85.72	87.93	**87.94**
5	74.87	76.63	80.1	83.3	87.91	85.81	85.72	**88.16**
6	74.78	76.11	80.05	83.6	87.24	85.77	87.66	**88.19**
7	74.51	76.07	79.92	83.6	88.6	85.41	87.86	**90.19**
8	73.98	76.31	79.62	83.49	88.33	86	87.94	**88.36**
9	74.07	75.91	79.86	83.44	88.08	85.7	87.78	**89.75**
10	74.03	76.02	79.57	83.63	88.11	85.74	87.25	**88.78**
Ave	74.44	76.3	79.86	83.54	87.95	85.73	87.61	**88.77**

Table 7
Ten rounds of holdout prediction results in terms of accuracy (%) for 2015–2016 (Period 2).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	78.03	79.72	80.39	79.87	81.79	80.33	85.88	**87.07**
2	77.17	78.57	79.62	78.15	81.35	78.47	84.27	**85.36**
3	78.93	79.70	80.93	80.45	82.69	80.45	85.42	**86.59**
4	77.51	78.24	79.24	78.63	81.26	79.20	84.38	**84.96**
5	78.13	79.80	80.79	79.68	82.00	80.08	85.59	**86.49**
6	78.20	79.53	80.55	80.12	82.12	80.49	85.25	**85.55**
7	78.30	79.16	80.55	79.55	82.27	79.57	85.02	**86.80**
8	78.67	79.53	80.55	79.84	82.41	80.55	85.96	**87.64**
9	78.32	79.34	80.24	79.97	81.97	80.47	85.57	**86.07**
10	77.21	78.13	78.95	78.68	80.91	79.11	85.63	**86.09**
Ave	78.05	79.17	80.18	79.49	81.88	79.87	85.30	**86.26**

[2] https://scikit-learn.org/stable
[3] https://xgboost.readthedocs.io
[4] https://github.com/tensorflow/tensorflow
[5] https://github.com/yunchuankong/forgeNet

Table 8
Ten rounds of holdout prediction results in terms of F1-score (%) for 2007–2014 (Period 1).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	84.39	88	89.63	90.17	92.69	92.82	92.88	**94.01**
2	84.55	88.43	90.01	90.74	92.54	92.94	92.78	**94.27**
3	84.76	88.17	89.9	90.58	92.95	93.09	93.4	**94.16**
4	84.45	87.89	89.62	90.51	92.82	92.95	92.94	**94.06**
5	84.67	88.38	89.98	90.42	93.24	92.93	93.56	**94.22**
6	84.68	88.18	89.91	90.42	92.45	93.01	92.79	**94.21**
7	84.68	88.05	89.82	90.64	93.24	92.79	93.05	**94.12**
8	84.33	88.25	89.78	90.44	92.93	93.04	93.1	**94.12**
9	84.45	87.98	89.85	90.52	92.98	92.89	92.83	**94.08**
10	84.55	88.46	89.71	90.35	92.14	90.55	92.82	**94.15**
Ave	84.55	88.18	89.82	90.48	92.80	92.70	93.02	**94.14**

Table 9
Ten rounds of holdout prediction results in terms of F1-score (%) for 2015–2016 (Period 2).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	74.61	77.6	78.23	79.3	81.03	80.93	91.14	**91.3**
2	75.31	76.12	78.99	76.9	80.9	77.25	89.38	**91.5**
3	78.89	78.49	81.13	81.68	82.88	81.68	90.66	**91.6**
4	76.63	75.43	78.53	78.47	81.97	80.23	89.65	**91.28**
5	76.43	78.21	80.47	79.16	81.68	80.34	91.02	**91.98**
6	76.31	76.78	79.7	80.38	81.3	81.92	90.09	**91.25**
7	75.7	75.75	79.6	77.95	81.88	78.63	89.83	**91.83**
8	78.09	78.43	80.49	80.26	83.42	82.88	92.58	**94.84**
9	76.75	76.97	79.36	80.52	82.01	82.16	91.19	**91.4**
10	76.68	76.88	78.91	79.62	82.12	81.28	92.17	**94.15**
Ave	76.54	77.07	79.54	79.42	81.92	80.73	90.77	**92.11**

Table 10
Ten rounds of holdout prediction results in terms of G-mean (%) for 2007–2014 (Period 1).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	72.54	75.89	79.77	82.71	87.47	86.17	88.05	**88.54**
2	72.47	76.75	80.6	83.71	87.4	86.57	88.52	**89.16**
3	72.57	75.93	79.81	83.42	88.05	86.72	88.66	**88.84**
4	72.42	75.77	79.9	83.31	87.87	86.61	87.83	**88.86**
5	72.9	76.39	80.34	83.05	88.51	86.57	88.38	**89.06**
6	72.88	75.85	80.3	83.29	87.37	86.64	87.39	**89.1**
7	72.54	75.65	80.07	83.39	88.51	86.18	88.78	**89.22**
8	72.22	76.32	80.08	83.38	88.25	86.94	88.4	**88.99**
9	72.14	75.59	80.14	83.26	88.05	86.51	88.75	**88.85**
10	72.46	76.21	80.05	83.15	88.21	86.88	88.57	**89.13**
Ave	72.51	76.04	80.11	83.27	87.97	86.58	88.33	**88.98**

The number of trees is in the range $[50, 300]$ with a step of 50. Except for the number of trees, the learning rate determines the contribution of each tree in XGBoost, and the search range for the learning rate is set from 0.001 to 0.1 with an equal ratio of 0.1. For DNN, the number of hidden layers is tested from 2 to 5, the number of hidden units is set from 2 to 6 with the step of 1, the learning rate is set from 10^{-5} to 10^{-1}, and the activate function includes Relu, Sigmoid and tanh.

In addition, to investigate the relationship between the prediction performance of the proposed model and the interaction strength among the features, a sensitivity analysis is conducted by testing the prediction performance of the proposed model with different numbers of trees for xgboost. The search range of the number of trees is also set from 50 to 500 with a step of 50.

Table 11
Ten rounds of holdout prediction results in terms of G-mean (%) for 2015–2016 (Period 2).

id	LR	SVM	RF	XGB	XGB+LR	DNN	XGB+DNN	XGB+forgeNet
1	73.81	75.54	75.77	76.3	76.92	77.17	80.71	**82.61**
2	73.91	74.51	75.98	74.81	76.68	75.01	79.83	**81.2**
3	76.01	75.84	77.22	77.52	77.8	77.52	80.87	**82.55**
4	74.56	74.16	75.7	75.6	76.93	76.51	79.93	**81.62**
5	74.66	75.68	76.85	76.15	76.82	76.77	80.47	**81.96**
6	74.59	74.97	76.43	76.76	77.4	77.56	79.91	**81.46**
7	74.31	74.45	76.38	75.51	77.44	75.84	80.48	**82.5**
8	75.45	75.71	76.78	76.62	77.7	78	81.25	**83.12**
9	74.8	75.03	76.24	76.8	76.69	77.67	80.93	**82.56**
10	74.51	74.77	75.82	76.11	77.29	76.97	80.74	**82.35**
Ave	74.66	75.07	76.32	76.22	77.17	76.90	80.51	**82.19**

at a relatively high position compared with the other individual classifiers.

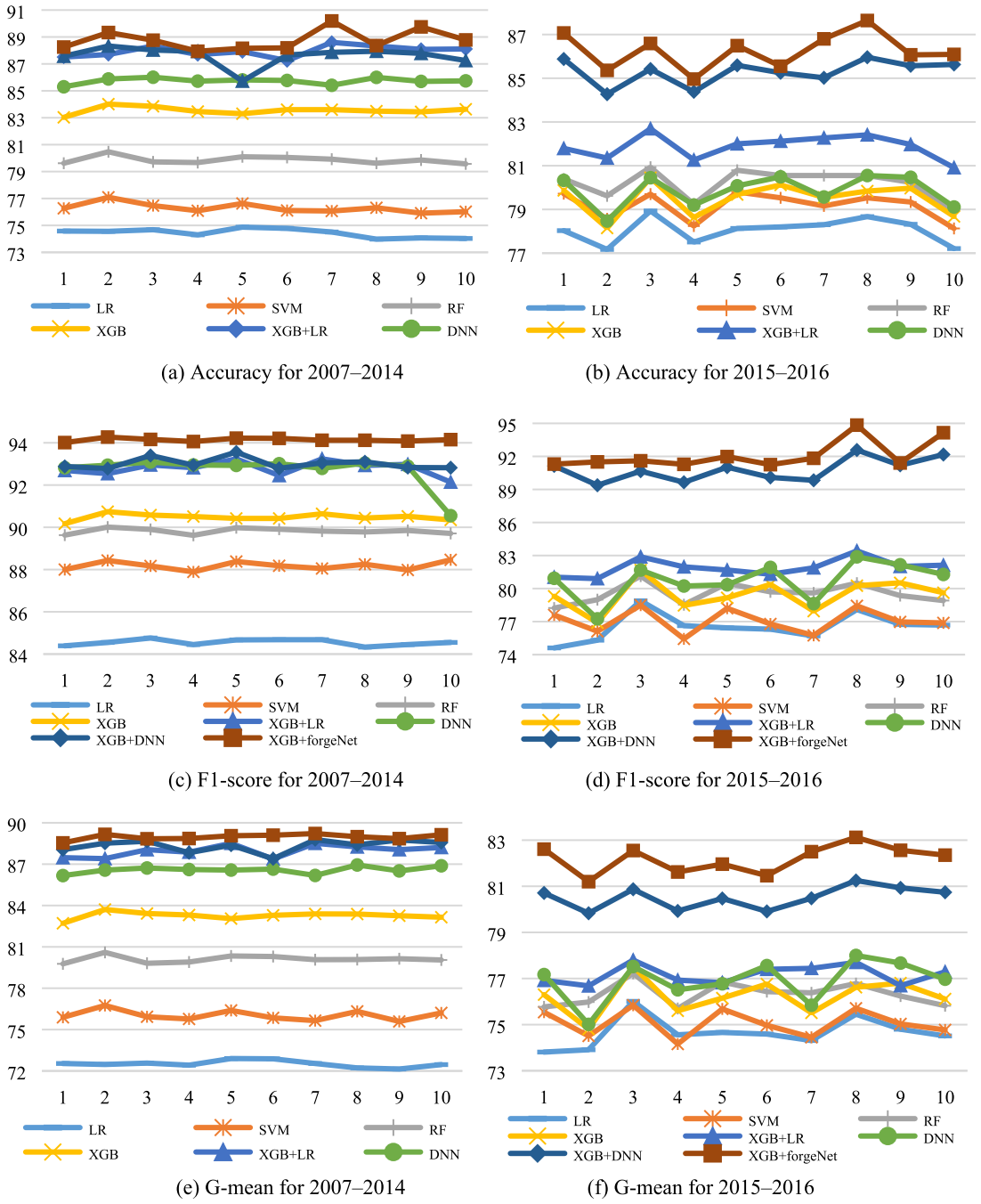


Fig. 6. Prediction performance for all classifiers.

5.1.3. Performance metrics

The confusion matrix is an important tool to evaluate the performance of classifiers, and various performance metrics are designed according to the confusion matrix. The row presents the prediction category, and the column presents the actual category. The confusion matrix can be presented as in Table 5:

where TP, FP, FN, and TN denote the true positive, false-positive, false negative and true negative, respectively.

To measure the prediction performance comprehensively, the accuracy, G-mean and F1-score are deployed for both the reference test and the unbalanced data test. The accuracy refers to the ratio between the correct prediction samples and the total samples, and it is defined as $acc = \frac{TP+TN}{TP+FP+FN+TN}$. The accuracy is affected by the unbalanced data. The G-mean and F1-score can be used to evaluate the prediction performance for unbalanced data. The G-mean refers to the square of acc^+ and acc^- and is defined as $G-mean = \sqrt{acc^+ \bullet acc^-} = \sqrt{\frac{TP}{TP+FN} \bullet \frac{TN}{TN+FP}}$. The F1-score is the harmonic mean of the precision and recall, and it is defined as $F_1-score = 2 \bullet \frac{\text{precision} \bullet \text{recall}}{\text{precision}+\text{recall}}$.

5.2. Numerical comparison

5.2.1. Comparison of the prediction results

The ten rounds of prediction results across two periods, including the

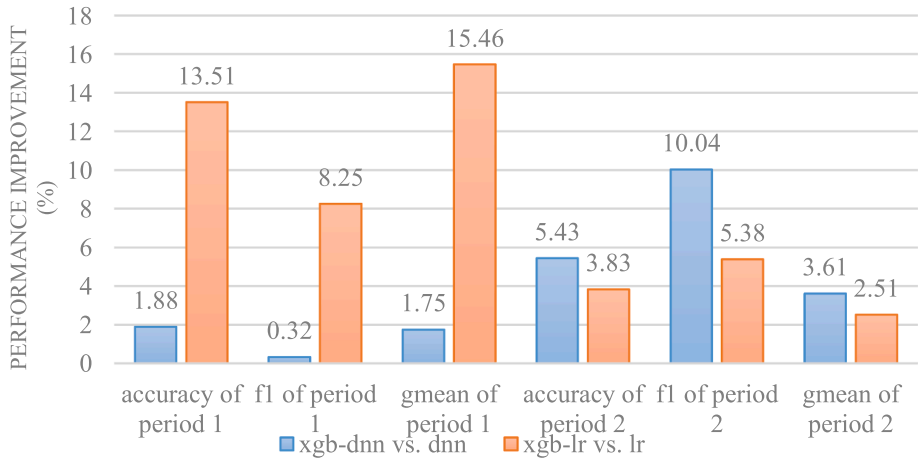


Fig. 7. Performance improvements of DNN and LR using XGB.

Table 12
Significance test results of three comparison groups with the Friedman test.

Model	Period	Compared Model(s)	Performance metric	p value	Significant level	Hypothesis
XGB + DNN	2007–2014, 2015–2016	DNN	Accuracy	0.00***	1%	H1
			F1-score	0.00***	1%	
			G-mean	0.00***	1%	
XGB + forgeNet	2007–2014	XGB + DNN	Accuracy	0.00***	1%	H2
			F1-score	0.00***	1%	
			G-mean	0.00***	1%	
	2015–2016		Accuracy	0.00***	1%	
			F1-score	0.00***	1%	
			G-mean	0.00***	1%	
XGB + forgeNet	2007–2014, 2015–2016	LR, SVM, RF, XGB	Accuracy	0.00***	1%	H3
			F1-score	0.00***	1%	
			G-mean	0.00***	1%	

accuracy, f1-score and Gmean, are listed in Tables 6–11. Five standalone classifiers and three hybrid classifiers, including

LR, SVM, RF, XGB, XGB + LR, DNN, XGB + DNN and XGB + forgeNet, are verified across two time periods. Tables 6, 8 and 10 present the prediction performance for the accuracy, f1-score and Gmean in the period 2007–2014, while Tables 7, 9 and 11 present the prediction performance for the accuracy, f1-score and Gmean in the period 2015–2016. The best prediction performance for each evaluation metric is marked in bold in each table.

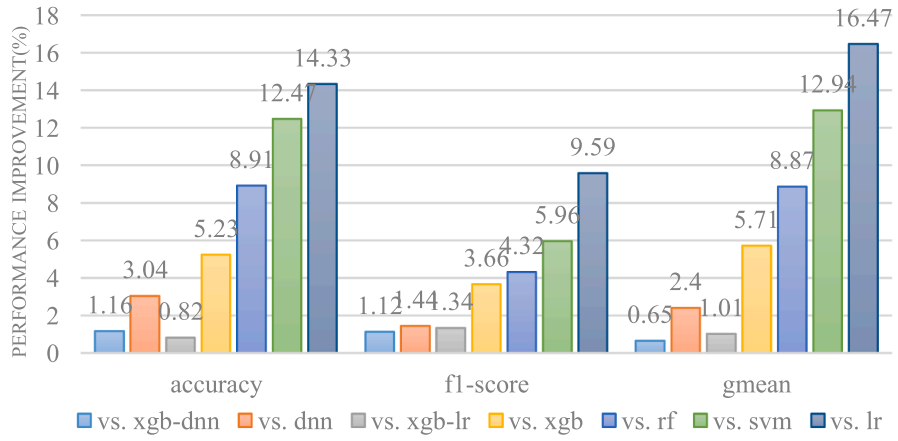
From the prediction results, we can derive several findings. First, for standalone classifiers, Tables 6-11 show that the average prediction accuracy of DNN is 85.73% and 79.87%, the f1-score of DNN is 92.7% and 80.73%, the G-mean of DNN is 86.58% and 76.9%, respectively, for the two time periods. From these results, DNN achieved the best prediction performance across two time periods and was superior to the other standalone classifiers in terms of the accuracy, F1-score and Gmean. Second, the prediction performances of the tree-based ensemble models, namely, RF and XGB, outperformed other single classifiers (LR and SVM). Third, although XGB is a type of ensemble model, DNN can obtain better prediction performance in credit risk prediction due to its feasible structure and outstanding nonlinear mapping ability. This finding indicated that DNN has the strongest ability in credit risk prediction tasks when addressing credit features with complex relationships. Fig. 6 also shows that the prediction curve of DNN is at a relatively high position compared with the other individual classifiers.

For hybrid classifiers, from Table 6-11, it can be observed that the prediction performances of LR and DNN are enhanced by using the transformed features to a certain extent. Specifically, the mean prediction accuracy was 87.95% and 81.88% for XGB + LR, and the mean prediction accuracy of XGB + DNN was 87.61% and 85.30%, respectively, across the two time periods. The prediction accuracy was enhanced by 13.51% (87.95–74.44) and 1.88% (87.61–85.73), corresponding to LR and DNN, after using the feature extraction technique in the time period of 2007–2014. Although XGB + DNN achieved a better prediction performance than XGB + LR, the improvement of LR is much larger than DNN after feature transformation by using XGB. The mean f1-scores of XGB + LR and XGB + DNN were 92.80% and 93.02% in the 2007–2014 time period, which were improved by 2.32% (92.80–90.48) and 0.32 (93.02–92.70), respectively. The mean values of Gmean of XGB + LR and XGB + DNN are 87.97% and 88.33, respectively, which are also improved to a certain extent. This finding suggests that XGB is a useful tool for feature linearization, and the transformed features are more separable than the original features; thus, the prediction results are improved by using the transformed feature set.

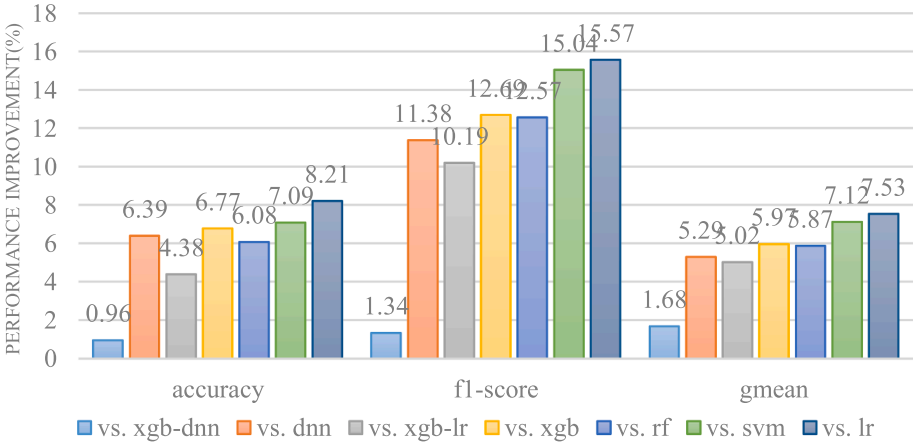
Fig. 6 shows that the prediction curve positions of the DNN and LR with XGB are higher than those of the DNN and LR without XGB. In addition, as shown in Fig. 7, regardless of the time period, the performance metrics for DNN and LR are enhanced to a certain extent by using feature linearization with XGB.

To verify the effectiveness of H1, a statistical significance analysis is conducted. As shown in Table 12, we can observe that the results of the hypothesis on the predictive performance between XGB + DNN and DNN indicated that the hypothesis is accepted regarding the three prediction performance metrics. This result shows that the prediction performance of XGB + DNN is significantly better than that of DNN due to the linearization of the original features by using the XGB feature transformation technique.

For the proposed two-stage hybrid model, the results show that XGB + forgeNet achieves first place among these models and obtains the best mean accuracy of 88.77% and 86.26% in two time periods. The performance of XGB + forgeNet on the other two metrics, F1-score and G-mean, also ranked first. In detail, the mean f1-score of XGB + forgeNet



(a) Performance improvement of XGB-forgeNet in the time period of 2007–2014



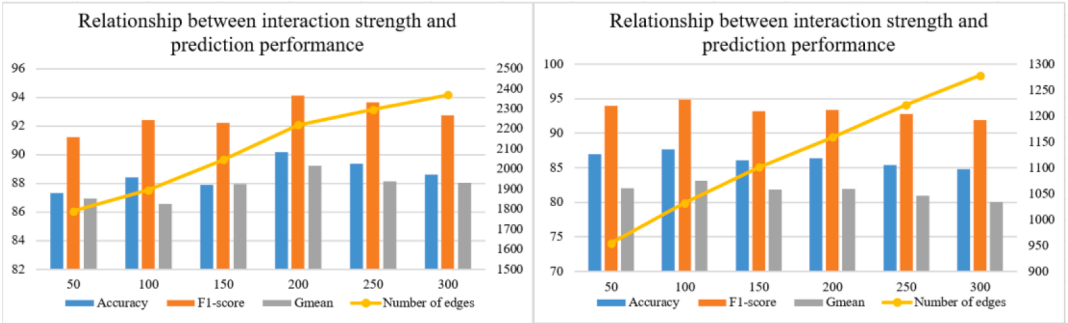
(b) Performance improvement of XGB-forgeNet in the time period of 2015–2016

Fig. 8. Performance improvement of XGB-forgeNet.

is 94.14% and 92.11%, and the mean values of G-mean for XGB + forgeNet are 88.98% and 82.19%, respectively. These results are obviously higher than those of the other benchmarks. The excellent performance of F1-score and G-mean also indicate that the proposed model is good at handling imbalanced credit datasets, which is an indispensable aspect of credit risk prediction tasks. The improvement in the prediction performance of the proposed model mainly contributes to two aspects: the initial improvement benefits from the first stage of the proposed two-stage model by using XGB to transform the original features. Then, feature graph mining of the transformed features further enhances the prediction performance of XGB + forgeNet.

As shown in Fig. 6, whether compared with the time dimension or from the metric dimension, the prediction curves of XGB-forgeNet are at the highest locations compared with other benchmarks in most cases. From Fig. 8 (a)(b), it can also be observed that the average prediction performances of XGB-forgeNet are improved by using feature linearization and feature graph mining techniques. The advantages of XGB-forgeNet are more obvious compared with the individual benchmarks.

To examine H2 and H3, statistical significance tests are conducted. As shown in Table 12, the significance test between XGB + DNN and



(a) 2007–2014 (b) 2015–2016

Fig. 9. Sensitivity analysis between interaction strength and prediction performance.

XGB + forgeNet indicates that the hypothesis is accepted. This result shows that the prediction performance of XGB + forgeNet is significantly better than that of XGB + DNN. In addition, the significance test between XGB + forgeNet and other benchmark individuals indicates that the hypothesis is accepted. This result shows that the prediction performance of XGB + forgeNet is significantly better than that of other individual methods. Thus, XGB + forgeNet is an efficient credit risk prediction model that is suitable for different economic environments.

5.2.2. Sensitivity analysis

To provide an easy-to-operate manual for the proposed model, we validate the prediction performance of the proposed model under different settings. In the experiment, we test the relationship between the interaction strength among the features and the prediction performance. Fig. 6 plots the relationship between the interaction strength and the prediction performance, where the x-axis presents the number of trees for feature graph mining, the y-axis presents the prediction performance of XGB + forgeNet under different numbers of trees, and the auxiliary y-axis presents the number of edges among features.

From these figures, we can observe that the prediction performances of XGB + forgeNet are different with different numbers of trees. The number of edges increases with the increase in trees for the ensemble. From Fig. 9(a), we can observe that when the number of trees for XGB is 200, the prediction performances achieve the best accuracy, F1-score and G-mean, which are 90.19%, 94.12% and 89.22%, respectively. The prediction performances decrease with an increasing number of trees. The reason is that as the number of trees increases, increasingly useless relationships among features are incorporated into the feature graph, which can further deteriorate the prediction performance of the DNN. When the number of trees is below 200, the prediction performances are improved with the increasing number of trees. A small number of trees restricts the interaction strengths among the features, and some hidden relationships among features might not be fully discovered; thus, the prediction performance can be enhanced by increasing the number of trees for XGB. Therefore, there is a trade-off between the interaction strength and the prediction performance of the proposed model. If there are many trees for the ensemble, the irrelevant relationships among features will deteriorate the prediction performance of the DNN. Otherwise, a small number of XGB trees will cause underfitting for the DNN due to the loss of some useful links among features.

5.3. Discussion

The comparison experimental results indicated that DNN is an excellent choice in credit risk prediction compared with other prevalent classifiers, including LR, SVM, RF, and XGBoost. However, because of the nonlinear topological property of the credit dataset and the complex interactions among the transformed features, there is still room for further improvement by considering feature linearization and feature graphs. Through the validation of H1, we can conclude that the feature linearization process can boost the prediction performance of LR and DNN. Hypothesis H2 has demonstrated that the prediction performance of DNN is improved by incorporating the feature graph into DNN. The results have proven the important role of feature interactions. In addition, the number of trees is one of the main parameters that determines the performance of the proposed model. This approach should consider the trade-off between interaction strength and prediction performance.

6. Conclusions and future work

In this study, we present a two-stage hybrid model using XGBoost and a graph-based deep neural network to improve the prediction performance for credit risk prediction. The proposed model mainly consists of two stages. The first stage is feature linearization by using XGBoost. XGBoost has proven its ability for feature transformation in credit risk tasks. The prediction results and the acceptance of the hypothesis showed that XGBoost is an effective tool in data preprocessing for credit risk prediction. In the second stage, we employ forgeNet to handle the complex relationships between features and to produce the prediction results. The feature graph is fed into the DNN for the final prediction. The proposed two-stage framework not only transforms the original features to ensure better linear separability but also conducts deep mining on feature relations, which improves the prediction performance of the DNN.

The empirical analysis shows that the proposed two-stage hybrid model achieved the best average prediction performance of 87.52%, 93.13% and 85.59% in terms of the accuracy, F1-score, and G-mean compared with other individuals and hybrid models. XGB + forgeNet achieved significant improvements of 6.14%, 7.59% and 6.18%, respectively, which suggests that the proposed model has good ability to address imbalanced datasets. The results of the significance test indicate that the advantages of the proposed two-stage hybrid model are mainly attributed to feature linearization and feature graph mining when utilizing DNN for credit risk prediction. Furthermore, the configuration of the number of trees for feature relation mining is conducted to investigate the relationships between feature interaction strengths and the prediction performance. A tuning strategy can be designed according to sensitivity analysis to improve the prediction performance of the proposed two-stage hybrid model.

The proposed two-stage hybrid model has three superiorities. First, the original features with nonlinear relationships are transformed to the more separable features, which can improve the prediction performance of the base classifier. Second, the relational information hidden in the transformed features is discovered by constructing a feature graph. Third, the DNN is improved by using a feature graph, which can further enhance the prediction performance of the credit risk.

However, there are some limitations and future research directions of this research: (1) this study only applied XGBoost for feature extraction and feature graph construction; since XGBoost is one type of tree-based ensemble model, more generalized scenarios, such as bagging tree and boosting tree models, should also be considered for handling these feature engineering processes in our future study; (2) the explanation and interpretation of the machine learning models are the main focus of intelligent diagnosis in financial industry applications, and it is necessary to reinforce the explanation ability of the proposed model; (3) more and more sophisticated machine learning models are applied in the classification and prediction tasks of the financial industry field, and it would be interesting to verify the application effect of the proposed model in other financial application scenarios in the future.

CRediT authorship contribution statement

Jiaming Liu: Conceptualization, Methodology, Software, Writing – review & editing. **Sicheng Zhang:** Data curation, Writing – original draft. **Haoyue Fan:** Visualization, Investigation.

Declaration of Competing Interest

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

Acknowledgement

We also acknowledge the support of the National Natural Science Foundation of China (NO. 71901014, 71901015), Beijing Social Science Foundation (NO.19GLC062).

References

BinSaeedan, W., & Alramlawi, S. (2021). CS-BPSO: Hybrid feature selection based on chi-square and binary PSO algorithm for Arabic email authorship analysis. *Knowledge-Based Systems, 227*, Article 107224. https://doi.org/10.1016/j.knosys.2021.107224

Chang, Y.-C., Chang, K.-H., & Wu, G.-J. (2018). Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. *Applied Soft Computing, 73*, 914–920. https://doi.org/10.1016/j.asoc.2018.09.029

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proc. 22nd ACM SIGKDD. *The International Conference on Knowledge Discovery and Data Mining., 785–794*. https://doi.org/10.1145/2939672.2939785

Cui, L., Bai, L., Wang, Y., Jin, X., & Hancock, E. R. (2021). Internet financing credit risk evaluation using multiple structural interacting elastic net feature selection. *Pattern Recognition, 114*, Article 107835. https://doi.org/10.1016/j.patcog.2021.107835

Cui, L., Bai, L., Wang, Y., Yu, P. S., & Hancock, E. R. (2021). Fused lasso for feature selection using structural information. *Pattern Recognition, 119*, Article 108058. https://doi.org/10.1016/j.patcog.2021.108058

Dastile, X., Celik, T., & Potsane, M. (2020). Statistical and machine learning models in credit scoring: A systematic literature survey. *Applied Soft Computing, 91*, Article 106263. https://doi.org/10.1016/j.asoc.2020.106263

Dumitrescu, E., Hué, S., Hurlin, C., & Tokpavi, S. (2021). Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research, S0377221721005695*. https://doi.org/10.1016/j.ejor.2021.06.053

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*, 1189–1232. https://doi.org/10.1214/aos/1013203451

Fu, S., Liu, W., Zhang, K., & Zhou, Y. (2021). Example-feature graph convolutional networks for semi-supervised classification. *Neurocomputing, 461*, 63–76. https://doi.org/10.1016/j.neucom.2021.07.048

Fu, X., Ouyang, T., Chen, J., & Luo, X. (2020). Listening to the investors: A novel framework for online lending default prediction using deep learning neural networks. *Information Processing and Management, 57*, Article 102236. https://doi.org/10.1016/j.ipm.2020.102236

Gokalp, O., Tasci, E., & Ugur, A. (2020). A novel wrapper feature selection algorithm based on iterated greedy metaheuristic for sentiment classification. *Expert Systems with Applications, 146*, Article 113176. https://doi.org/10.1016/j.eswa.2020.113176

Golbayani, P., Florescu, I., & Chatterjee, R. (2020). A comparative study of forecasting corporate credit ratings using neural networks, support vector machines, and decision trees. *The North American Journal of Economics and Finance, 54*, Article 101251. https://doi.org/10.1016/j.najef.2020.101251

Haq, A. U., Zeb, A., Lei, Z., & Zhang, D. (2021). Forecasting daily stock trend using multi-filter feature selection and deep learning. *Expert Systems with Applications, 168*, Article 114444. https://doi.org/10.1016/j.eswa.2020.114444

He, X., Bowers, S., Candela, J. Q., Pan, J., Jin, O., Xu, T., … Herbrich, R. (2014). In *Practical Lessons from Predicting Clicks on Ads at Facebook, in* (pp. 1–9). New York, NY, USA: ACM Press. https://doi.org/10.1145/2648584.2648589.

Huang, Y., & Kou, G. (2014). A kernel entropy manifold learning approach for financial data analysis. *Decision Support Systems, 64*, 31–42. https://doi.org/10.1016/j.dss.2014.04.004

Jadhav, S., He, H., & Jenkins, K. (2018). Information gain directed genetic algorithm wrapper feature selection for credit rating. *Applied Soft Computing, 69*, 541–553. https://doi.org/10.1016/j.asoc.2018.04.033

Jiang, H. (2018). Model forecasting based on two-stage feature selection procedure using orthogonal greedy algorithm. *Applied Soft Computing, 63*, 110–123. https://doi.org/10.1016/j.asoc.2017.11.047

Jiang, L., Yu, G., Guo, M., & Wang, J. (2020). Feature selection with missing labels based on label compression and local feature correlation. *Neurocomputing, 395*, 95–106. https://doi.org/10.1016/j.neucom.2019.12.059

Jiménez-Cordero, A., Morales, J. M., & Pineda, S. (2021). A novel embedded min-max approach for feature selection in nonlinear Support Vector Machine classification. *European Journal of Operational Research, 293*, 24–35. https://doi.org/10.1016/j.ejor.2020.12.009

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y., 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree, in: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 3146–3154.

Kim, E., Lee, J., Shin, H., Yang, H., Cho, S., Nam, S., … Kim, J. (2019). Champion-challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning. *Expert Systems with Applications, 128*, 214–224. https://doi.org/10.1016/j.eswa.2019.03.042

Kim, K. (2016). A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree. *Pattern Recognition, 60*, 157–163. https://doi.org/10.1016/j.patcog.2016.04.016

Kong, Y., & Yu, T. (2020). forgeNet: A graph deep neural network model using tree-based ensemble classifiers for feature graph construction. *Bioinformatics, 36*, 3507–3515. https://doi.org/10.1093/bioinformatics/btaa164

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

Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with the Boruta Package. *Journal of Statistical Software, 36*. https://doi.org/10.18637/jss.v036.i11

Lappas, P. Z., & Yannacopoulos, A. N. (2021). A machine learning approach combining expert knowledge with genetic algorithms in feature selection for credit risk assessment. *Applied Soft Computing, 107*, Article 107391. https://doi.org/10.1016/j.asoc.2021.107391

Liang, L., & Cai, X. (2020). Forecasting peer-to-peer platform default rate with LSTM neural network. *Electronic Commerce Research and Applications, 43*, Article 100997. https://doi.org/10.1016/j.elerap.2020.100997

Lucas, Y., Portier, P.-E., Laporte, L., He-Guelton, L., Caelen, O., Granitzer, M., & Calabretto, S. (2020). Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs. *Future Gener. Comput. Syst., 102*, 393–402. https://doi.org/10.1016/j.future.2019.08.029

Luo, J., Yan, X., & Tian, Y. (2020). Unsupervised quadratic surface support vector machine with application to credit risk assessment. *European Journal of Operational Research, 280*, 1008–1017. https://doi.org/10.1016/j.ejor.2019.08.010

Ma, X., Sha, J., Wang, D., Yu, Y., Yang, Q., & Niu, X. (2018). Study on a prediction of P2P network loan default based on the machine learning LightGBM and XGboost algorithms according to different high dimensional data cleaning. *Electronic Commerce Research and Applications, 31*, 24–39. https://doi.org/10.1016/j.elerap.2018.08.002

Moghanian, S., Saravi, F. B., Javidi, G., & Sheybani, E. O. (2020). GOAMLP: Network Intrusion Detection With Multilayer Perceptron and Grasshopper Optimization Algorithm. *IEEE Access, 8*, 215202–215213. https://doi.org/10.1109/ACCESS.2020.3040740

Mokhtia, M., Eftekhari, M., & Saberi-Movahed, F. (2021). Dual-manifold regularized regression models for feature selection based on hesitant fuzzy correlation. *Knowledge-Based Systems, 229*, Article 107308. https://doi.org/10.1016/j.knosys.2021.107308

Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E.B., Turaga, D., 2017. Learning Feature Engineering for Classification, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence. Presented at the Twenty-Sixth International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence Organization, Melbourne, Australia, pp. 2529–2535. https://doi.org/10.24963/ijcai.2017/352.

Orsenigo, C., & Vercellis, C. (2013). Linear versus nonlinear dimensionality reduction for banks' credit rating prediction. *Knowledge-Based Systems, 47*, 14–22. https://doi.org/10.1016/j.knosys.2013.03.001

Paul, D., Jain, A., Saha, S., & Mathew, J. (2021). Multi-objective PSO based online feature selection for multi-label classification. *Knowledge-Based Systems, 222*, Article 106966. https://doi.org/10.1016/j.knosys.2021.106966

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., Gulin, A., n.d. CatBoost: unbiased boosting with categorical features 11.

Rtayli, N., & Enneya, N. (2020). Enhanced credit card fraud detection based on SVM-recursive feature elimination and hyper-parameters optimization. *J. Inf. Secur. Appl., 55*, Article 102596. https://doi.org/10.1016/j.jisa.2020.102596

Shen, F., Zhao, X., Kou, G., & Alsaadi, F. E. (2020). A new deep learning ensemble credit risk evaluation model with an improved synthetic minority oversampling technique. *Applied Soft Computing, 106852*. https://doi.org/10.1016/j.asoc.2020.106852

Shen, F., Zhao, X., Li, Z., Li, K., & Meng, Z. (2019). A novel ensemble classification model based on neural networks and a classifier optimisation technique for imbalanced credit risk evaluation. *Physica A: Statistical Mechanics and its Applications, 526*, Article 121073. https://doi.org/10.1016/j.physa.2019.121073

Sigrist, F., & Hirnschall, C. (2019). Grabit: Gradient tree-boosted Tobit models for default prediction. *Journal of Banking & Finance, 102*, 177–192. https://doi.org/10.1016/j.jbankfin.2019.03.004

Tarkhaneh, O., Nguyen, T. T., & Mazaheri, S. (2021). A novel wrapper-based feature subset selection method using modified binary differential evolution algorithm. *Information Sciences, 565*, 278–305. https://doi.org/10.1016/j.ins.2021.02.061

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. *J. Bus. Res., 130*, 200–209. https://doi.org/10.1016/j.jbusres.2021.03.018

Wang, L., & Wu, C. (2017). Business failure prediction based on two-stage selective ensemble with manifold learning algorithm and kernel-based fuzzy self-organizing map. *Knowledge-Based Systems, 121*, 99–110. https://doi.org/10.1016/j.knosys.2017.01.016

Xia, Y., Zhao, J., He, L., Li, Y., & Niu, M. (2020). A novel tree-based dynamic heterogeneous ensemble method for credit scoring. *Expert Systems with Applications, 159*, Article 113615. https://doi.org/10.1016/j.eswa.2020.113615

Yu, L., Yao, X., Zhang, X., Yin, H., & Liu, J. (2020). A novel dual-weighted fuzzy proximal support vector machine with application to credit risk analysis. *Int. Rev. Financ. Anal., 71*, Article 101577. https://doi.org/10.1016/j.irfa.2020.101577

Zhang, X., Han, Y., Xu, W., & Wang, Q. (2019). HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture. *Information Sciences, S002002551930427X*. https://doi.org/10.1016/j.ins.2019.05.023

Zhang, Y., Zhang, Z., Qin, J., Zhang, L., Li, B., & Li, F. (2018). Semi-supervised local multi-manifold Isomap by linear embedding for feature extraction. *Pattern Recognition, 76*, 662–678. https://doi.org/10.1016/j.patcog.2017.09.043

Zhou, H., Zhang, J., Zhou, Y., Guo, X., & Ma, Y. (2021). A feature selection algorithm of decision tree based on feature weight. *Expert Systems with Applications, 164*, Article 113842. https://doi.org/10.1016/j.eswa.2020.113842