ARTICLE IN PRESS

International Journal of Forecasting xxx (xxxx) xxx

Contents lists available at ScienceDirect

International Journal of Forecasting

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

A novel deep ensemble model for imbalanced credit scoring in internet finance

Jin Xiao [a,b,*], Yu Zhong [a], Yanlin Jia [c], Yadong Wang [d], Ruoyi Li [e], Xiaoyi Jiang [f], Shouyang Wang [g,**]

[a] Business School, Sichuan University, Chengdu, 610065, China
[b] Institute of Management Science and Operations Research, Sichuan University, Chengdu, 610065, China
[c] School of Sciences, Southwest Petroleum University, Chengdu, 610500, China
[d] Pubilc Administration School, Guangzhou University, Guangzhou, Guangdong, 510006, China
[e] Harris School of Public Policy, University of Chicago, Chicago, 60637, USA
[f] Faculty of Mathematics and Computer Science, University of Münster, Münster, 48149, Germany
[g] School of Entrepreneurship and Management, ShanghaiTech University, Shanghai, 201210, China

ARTICLE INFO

Keywords:
Credit scoring
Deep ensemble
Class imbalance
VAE
Deep forest

ABSTRACT

Most existing deep ensemble credit scoring models have considered deep neural networks, for which the structures are difficult to design and the modeling results are difficult to interpret. Moreover, the methods of dealing with the class-imbalance problem in these studies are still based on traditional resampling methods. To fill these gaps, we combine a new over-sampling method, the variational autoencoder (VAE), and a deep ensemble classifier, the deep forest (DF), and propose a novel deep ensemble model for credit scoring in internet finance, VAE–DF. We train and test our model using a number of credit scoring datasets in internet finance and find that our model exhibits good performance and can realize a self-adapting depth. The results show that VAE–DF is an effective credit scoring tool, especially for highly class-imbalanced and non-linear datasets in internet finance, due to its strong ability to learn the complex distributions of these datasets.

1. Introduction

With the rapid development of internet finance, credit risk management has become an important task for the internet finance industry. In general, credit scoring provides an effective tool for solving this problem (Thomas, 2000). Credit scoring is a binary classification problem in which loan applicants are distinguished into two classes, good credit and bad credit, based on characteristics such as loan amount, salary, and occupation (Durand, 1941).

Unlike traditional credit scoring data used by banks, the credit scoring data typically applied in internet finance are characterized by a high number of dimensions, large size, and multi-source heterogeneity (Li et al., 2018). Consequently, deep learning methods based on deep neural networks (DNNs) have been used to construct credit scoring models for internet finance (Tan et al., 2019; Wang et al., 2018, 2022; Zhu et al., 2018). However, DNNs have some drawbacks such as the difficulty of designing the structures and their black-box nature. To solve this problem, the deep forest (DF) (Zhou & Feng, 2019) is an interesting alternative. The DF is a deep ensemble method based on random forests (RF) that can realize a self-adapting depth.

The presence of noise and errors in the datasets is another common problem for credit scoring. García et al. (2012) used instance selection methods to remove the

* Corresponding author at: Business School, Sichuan University, Chengdu, 610065, China.
** Corresponding author.
E-mail addresses: xjxiaojin@126.com (J. Xiao), sywang@shanghaitech.edu.cn (S. Wang).

https://doi.org/10.1016/j.ijforecast.2023.03.004
0169-2070/

Please cite this article as: J. Xiao, Y. Zhong, Y. Jia et al., A novel deep ensemble model for imbalanced credit scoring in internet finance. International Journal of Forecasting (2023), https://doi.org/10.1016/j.ijforecast.2023.03.004.

noisy samples in training sets and found that the filtered sets performed significantly better than the non-preprocessed sets. Crone and Finlay (2012) explored a variety of sampling strategies and the results showed that using larger samples in credit scoring practice provided a significant increase in accuracy. Liu and Pan (2018) proposed a new hybrid classifier for credit scoring based on fuzzy-rough instance selection, and the new model had better classification accuracy on two credit scoring datasets. Tsai et al. (2021) used instance selection to deal with the noise problem and found that this method could improve the prediction performance of credit scoring models.

Apart from high dimensionality and the noise problem, class imbalance is an important problem in credit scoring. Class imbalance occurs insofar as most loan applicants fall into the good credit class, and the remaining (relatively) few applicants fall into the bad credit class (Xiao et al., 2021). In the context of internet finance, this problem may be even more serious than in other areas of finance. Consider the Lending Club,[1] the largest US P2P loan platform, as an example: its first-quarter data in 2017 show that bad credit applicants accounted for only 0.76% of all applicants. In fact, imbalanced data will seriously affect the classification performance of credit scoring models (Crone & Finlay, 2012). For example, for a credit scoring dataset where only 1% of the samples belong to the minority class (bad credit), even if a model classifies all samples as the majority class (good credit), it still achieves an overall accuracy of 99%. In spite of the high overall accuracy, all of the minority class samples, which we most want to classify accurately, are misclassified.

In deep learning applications, one of the most popular methods for addressing the class-imbalance problem is over-sampling (Levi & Hassner, 2015).[2] The synthetic minority over-sampling technique (SMOTE) (Chawla et al., 2002) is commonly used for credit scoring (Shen et al., 2021). However, SMOTE and its variants[3] are based on a linear interpolation approach and do not consider the feature distribution of minority class samples. In high-dimensional space, the new minority class samples generated by these approaches are likely to deviate from the original distribution, which may introduce external noise and lead to classification bias towards the majority class samples (Blagus & Lusa, 2012).

Therefore, some generative models that can generate complex and high-dimensional data are used for over-sampling to generate minority class samples. One of the most common methods is the generative adversarial network (GAN) (Douzas & Bacao, 2018; Engelmann & Lessmann, 2021; Fiore et al., 2019). The variational autoencoder (VAE) (Kingma & Welling, 2013) is another common generative model, although it is rarely used for over-sampling. Moreover, most studies focus on unstructured data such as image data, and the VAE has not yet been used for credit scoring. Therefore, this study aimed at using the VAE for over-sampling to generate high-dimensional credit scoring data in internet finance.

To this end, we propose a novel deep ensemble model for credit scoring in internet finance by combining the VAE and DF methods. First, the VAE learns the feature distribution of minority class samples and generates new minority class samples. Next, the deep ensemble DF classifier is trained using the balanced training set. Experiments are carried out on some credit scoring datasets in internet finance.

In summary, we make the following contributions. First, to the best of our knowledge, this is the first attempt to use the VAE to solve the imbalanced-data problem in credit scoring and improve the credit scoring performance. Second, we introduce the DF into the modeling of credit scoring for internet finance and propose a novel deep ensemble model, VAE–DF, thus extending existing research on deep ensemble credit scoring models. Third, we demonstrate that the VAE outperforms random over-sampling (ROS) and SOMTE-style methods when DF is used as the classifier, especially on highly class-imbalanced and non-linear credit scoring datasets. Fourth, we propose a new method of calculating the feature importance for the VAE–DF model.

The remainder of this paper is structured as follows. Section 2 presents the literature review. In Section 3, we briefly review the VAE algorithm and DF classifier, and then introduce the basic idea and modeling steps of the VAE–DF model. In Section 4, we describe the credit scoring datasets for internet finance and the data preprocessing procedure. In Section 5, we describe the experimental setup and the evaluation measures, and then analyze the experimental results. Finally, Section 6 presents our conclusions from this study and proposes possible directions for future research.

2. Literature review

2.1. Deep ensemble models for credit scoring

Because deep learning methods are superior to traditional machine learning methods and ensemble classifiers perform better than single classifiers, deep ensemble models based on DNNs have recently been constructed for credit scoring and achieved better classification performance than some benchmark credit scoring models. For example, Yu et al. (2016) proposed a multistage deep belief network (DBN) based on an extreme learning machine (ELM) for credit scoring. They utilized the ELM as the single classifier and the DBN model to get the final classification results. The results showed the superiority of this new model in terms of high overall accuracy. Khan et al. (2020) applied four single classifiers—logistic regression (LR), decision trees, naive Bayes, and a DBN—to credit scoring datasets and then used ensemble learning to get the final result. They found that the proposed model performed better than other single models.

[1] See https://www.lendingclub.com for more information.

[2] Over- and under-sampling are resampling methods, which alter the original class distribution of imbalanced data by increasing minority class samples or eliminating majority class samples. Marqués et al. (2013) demonstrated that over-sampling outperforms under-sampling in most cases. Therefore, we focus on the over-sampling method and adopt it to solve the class-imbalance problem in this paper.

[3] Both borderline-SMOTE (Han et al., 2005) and adaptive synthetic sampling (ADASYN) (He et al., 2008) are variants of SMOTE.

However, these studies did not consider the class-imbalance problem and measured the classification performance by overall accuracy, which is less suitable in imbalanced learning. Shen et al. (2021) developed a new deep learning ensemble credit scoring model to deal with imbalanced data. They proposed an improved SMOTE over-sampling method and combined a long short-term memory (LSTM) network and adaptive boosting algorithm to construct a new credit scoring model. The experimental results indicated that the proposed model was generally more competitive when addressing imbalanced credit scoring problems than benchmark models. However, this new method is still based on SMOTE, so that it may have similar drawbacks to the conventional SMOTE approach.

2.2. GAN-based over-sampling for imbalanced tabular data

Several studies focus on the GAN-based over-sampling method for imbalanced tabular data. Fiore et al. (2019) trained a standard GAN on minority class samples to generate new minority class samples. They performed experiments on a credit scoring dataset with only numerical variables and found that their method achieved an improved sensitivity measure compared with SMOTE. Douzas and Bacao (2018) proposed the use of a conditional GAN (cGAN) as an over-sampling method. They evaluated their method on datasets with numerical features using five different classifiers. The results showed that cGAN performed better than traditional over-sampling methods for a variety of classifiers, evaluation metrics, and datasets. Engelmann and Lessmann (2021) proposed a conditional Wasserstein GAN (cWGAN)-based over-sampling method of tabular data for imbalanced learning. They took empirical comparisons in the credit scoring context and the results showed the competitiveness of the method compared to standard over-sampling methods. They also found that complex data were challenging for any over-sampling method, but that cWGAN could master the challenge better than traditional over-sampling methods. Our study differs from Engelmann and Lessmann (2021) in two ways. First, our resampling methods are different. We use a VAE for over-sampling whereas their method is based on cWGAN. Second, we employ the DF as the classifier and propose a deep ensemble credit model, VAE–DF. They focused on the general over-sampling performance of cWGAN for standard classification algorithms such as LR and random forests (RF).

2.3. VAE-based over-sampling for imbalanced data

Apart from the GAN, the VAE is another promising deep generative model of unsupervised learning for complicated distributions. The VAE has already shown promise in generating many kinds of complex data, such as handwritten digits (Salimans et al., 2015), face recognition data (Rezende et al., 2014), and physical scenes (Kulkarni et al., 2015). In particular, the VAE can be used for over-sampling by learning the feature distribution of the minority class samples, thus achieving better performance than traditional over-sampling methods. For example, Wan et al. (2017) employed a VAE-based method for over-sampling to solve image classification problems and the results showed that the new method outperformed traditional over-sampling methods, including SMOTE and ADASYN, under various evaluation metrics. However, their model is constructed using image data. Only a few studies focus on VAE-based generation of tabular data. Zhang et al. (2018) employed a VAE for over-sampling to generate tabular data. However, they did not research credit scoring problems and only used continuous features as the final set to train the VAE network.

In addition, Li et al. (2019) used a VAE to generate synthetic data for privacy-preserving data sharing. They used one-hot encoding to convert categorical features. The results showed that the VAE can be given to a data user to generate customized data that closely mimic the original dataset.

In summary, according to our literature review, most existing deep ensemble credit scoring models are based on DNNs, whose structure is difficult to design. For example, hyperparameters, such as the number of layers and neurons in each layer, need to be fixed before training, but they are difficult to pre-define and require considerable training (Ma et al., 2020; Xiao et al., 2019). Moreover, only one study has considered the class-imbalance problem for deep ensemble credit scoring models, with an improved SMOTE over-sampling method adopted to overcome this problem (Shen et al., 2021). Therefore, we consider the VAE method for over-sampling and combine it with a DF that realizes a self-adapting depth, and we construct a novel deep ensemble model, VAE–DF, for credit scoring in internet finance.

3. Variational autoencoder–deep forest model

3.1. VAE algorithm

The VAE is a deep generative method based on variational Bayesian inference theory. In this study, the VAE is used as an over-sampling method to generate new minority class samples. Let $X_{\min} = \{x_1, x_2, \ldots, x_{N_{\min}}\}(i = 1, 2, \ldots, N_{\min})$ be a set of minority class samples, where $x_i \in \mathbb{R}^d$, and d denotes the number of features for a customer sample. In this case, $X_{\min}$ can be regarded as a dataset consisting of $N_{\min}$ minority class samples of a continuous variable x, where we assume that the variable x is generated by a latent variable z, which is low-dimensional and normally distributed. The VAE is usually composed of an encoder and a decoder. The encoder is used to obtain the hidden variable z from the input $X_{\min}$, while the decoder is used to generate a new set of minority class samples $X'_{\min}$ by sampling z. The objective of the VAE is to make the distribution of $X'_{\min}$ consistent with that of $X_{\min}$. However, the real posterior distribution $p_\theta(z|x)$ is intractable; thus, the VAE introduces $q_\phi(z|x)$ to approximate $p_\theta(z|x)$. We denote $q_\phi(z|x)$ as the encoder (recognition model) and $p_\theta(x|z)$ as the decoder (generative model), where ϕ and θ are the parameters of the encoder and decoder, respectively.

To maximize the VAE objective function, we maximize the sum over the marginal likelihoods of all the minority class samples. The marginal likelihood for each minority class sample x_i ($i = 1, 2, \ldots, N_{\min}$) is $\log p_\theta(x_i) = \log \int p_\theta(x_i|z) p_\theta(z)\, dz$. By applying Bayes' formula and introducing the posterior distribution $p_\theta(z|x_i)$ and its approximation $q_\phi(z|x_i)$, the marginal likelihood for each minority class sample can be written as

$$\log p_\theta(x_i) = D_{KL}\left[q_\phi(z|x_i) \parallel p_\theta(z|x_i)\right] + \mathcal{L}(\theta, \phi; x_i). \quad (1)$$

The detailed derivation of Eq. (1) is given in Appendix A.1. In particular, the first term of Eq. (1) is the Kullback–Leibler (KL) divergence,[4] which measures the difference between $q_\phi(z|x_i)$ and $p_\theta(z|x_i)$. As the KL divergence is non-negative, $\log p_\theta(x_i) \geq \mathcal{L}(\theta, \phi; x_i)$. The second term $\mathcal{L}(\theta, \phi; x_i)$ is called the variational lower bound on the marginal likelihood $\log p_\theta(x_i)$. Based on conditional probability theory, $\mathcal{L}(\theta, \phi; x_i)$ is given by

$$\begin{aligned}\mathcal{L}(\theta, \phi; x_i) = &- D_{KL}\left[q_\phi(z|x_i) \parallel p_\theta(z)\right] \\ &+ \mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right]. \end{aligned} \quad (2)$$

The detailed derivation of Eq. (2) is described in Appendix A.2. Similarly, the first term of Eq. (2), $D_{KL}\left[q_\phi(z|x_i) \parallel p_\theta(z)\right]$, is the KL divergence measuring the difference between $q_\phi(z|x_i)$ and $p_\theta(z)$. Assuming that $p_\theta(z) = N(z; 0, I)$ and $q_\phi(z|x) = N(z; \mu, \sigma^2)$, we obtain the following equation:

$$\begin{aligned}&- D_{KL}\left[q_\phi(z|x_i) \parallel p_\theta(z)\right] \\ &= \frac{1}{2}\sum_{j=1}^{J}\left[\log\left(\sigma_i^j\right)^2 - \left(\mu_i^j\right)^2 - \left(\sigma_i^j\right)^2 + 1\right], \end{aligned} \quad (3)$$

where μ_i^j and σ_i^j denote the jth element of the mean vector μ_i and variance vector σ_i, respectively, and J denotes the dimensionality of the latent variable z. Here, both μ_i and σ_i are outputs of the encoder part of the VAE. The detailed derivation of Eq. (3) is given in Appendix A.3.

The second term $\mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right]$ in Eq. (2) is a log-likelihood expectation. By the reparameterization method,[5] the ith sample z_i of the latent variable z is obtained by the sampling process $z_i = \mu_i + \sigma_i \bigodot \varepsilon_i$, where $\varepsilon_i \sim N(0, I)$, and $\bigodot$ indicates the element-wise product. $\mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right]$ is then expressed as

$$\mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right] \approx \log p_\theta(x_i|z_i). \quad (4)$$

Substituting Eqs. (3) and (4) into Eq. (2), we obtain the final VAE objective function:

$$\begin{aligned}\mathcal{L}(\theta, \phi; x_i) \approx &\frac{1}{2}\sum_{j=1}^{J}\left[\log\left(\sigma_i^j\right)^2 - \left(\mu_i^j\right)^2 - \left(\sigma_i^j\right)^2 + 1\right] \\ &+ \log p_\theta(x_i|z_i). \end{aligned} \quad (5)$$

Fig. 1 shows a schematic diagram of the VAE network, where the encoder and decoder are MLPs with one hidden layer.[6] In the encoder, $h_1 = \tanh(W_1 x + b_1)$, $\mu = \tanh(W_2 h_1 + b_2)$, and $\log \sigma^2 = \tanh(W_3 h_1 + b_3)$. In the decoder, $z = \mu + \sigma \bigodot \varepsilon$, $h_2 = \tanh(W_4 z + b_4)$, and $x' = \text{sigmoid}(W_5 h_2 + b_5)$. Hence, $\phi = \{W_1, W_2, W_3, b_1, b_2, b_3\}$ and $\theta = \{W_4, W_5, b_4, b_5\}$ are the parameters of the encoder and the decoder, respectively, which can be obtained through the gradient descent algorithm.

3.2. Deep forest

The DF method is a supervised deep ensemble model inspired by DNNs and ensemble learning (Zhou & Feng, 2019). Current deep learning models are mostly built upon DNNs, which require significant effort in hyperparameter tuning. So it is necessary to explore deep learning models beyond DNNs. The DF proposed by Zhou and Feng (2019) has a cascade structure that enables representation learning. Its representational learning ability can be further enhanced by multi-grained scanning when the inputs have high dimensionality, potentially enabling DF to deal with image, audio, or textual data. Their experiment results show that the DF achieves highly competitive performance to DNNs. A key characteristic of deep learning is data-driven feature learning. In DNNs, this is realized by layer-by-layer in-model feature transformation. The DF is designed to hold these characteristics and thus represents another technique of deep learning. Zhou and Feng (2019) offer a detailed analysis of the most essential characteristics of DNNs and their corresponding realization in DF. Compared with DNNs, the DF has a number of appealing properties, including fewer hyperparameters and a data-dependent determination of model complexity.

The overall architecture of the DF used in this study is depicted in Fig. 2. Since credit scoring data are tabular, we do not introduce the multi-grained scanning mechanism in our paper. A DF has a self-adapting depth and multiple layers. Suppose there are L layers, each consisting of M forests, with each forest composed of T decision trees. Note that both M and T are hyperparameters.

Given a training sample x_i, each forest will first produce a class vector, as illustrated in Fig. 3. Take the mth ($m \in \{1, 2, \ldots, M\}$) forest in layer l ($l \in \{1, 2, \ldots, L\}$) as an example. First, the class vector of the tth ($t \in \{1, 2, \ldots, T\}$) tree in this forest is given by

$$P^{(l,m,t)}(x_i) = \left(P_0^{(l,m,t)}(x_i), P_1^{(l,m,t)}(x_i)\right), \quad (6)$$

where $P_0^{(l,m,t)}(x_i)$ and $P_1^{(l,m,t)}(x_i)$ denote the probabilities of classifying sample x_i into class 0 and class 1, respectively. Second, the class vector of the mth forest is obtained by averaging the class vector across all trees:

$$P^{(l,m)}(x_i) = \left(P_0^{(l,m)}(x_i), P_1^{(l,m)}(x_i)\right), \quad (7)$$

[4] The KL divergence measures the difference between two probability distributions $p(x)$ and $q(x)$. The KL divergence is calculated as $D_{KL}(q(x) \parallel p(x)) = \int q(x) \log \frac{q(x)}{p(x)} dx$. Note that $D_{KL}(q(x) \parallel p(x)) \neq D_{KL}(p(x) \parallel q(x))$. See Kullback (1997) for further information of the KL divergence.

[5] For further details on the reparameterization method, see Kingma and Welling (2013).

[6] For the encoder and decoder of the VAE, we could select other types of neural networks besides an MLP, depending on the data type. For simplicity and convenience, we adopt an MLP with only one hidden layer for illustration, as shown in Fig. 1. In the following experiments, to improve the performance of the VAE–DF models, we consider several MLPs with multiple hidden layers for the encoder and decoder.

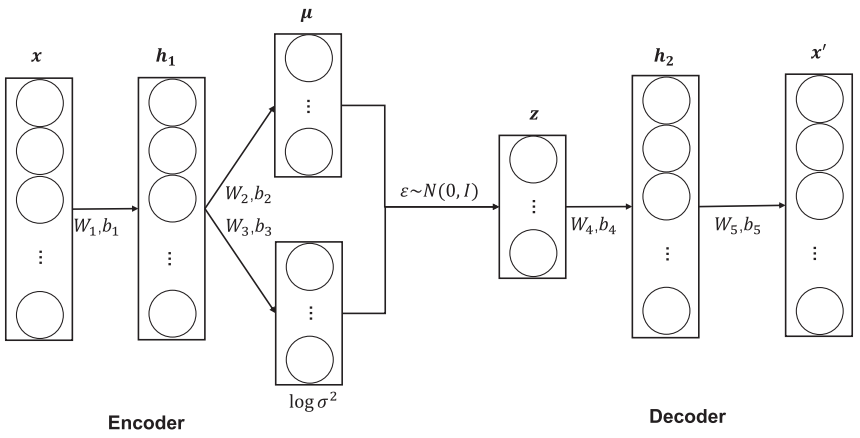


Fig. 1. Encoder and decoder of the VAE. Following Kingma and Welling (2013), we choose to fit $\log\sigma^2$ rather than σ^2 in the training process. An activation function is needed to transfer the value of σ^2, since it is always non-negative. However, $\log\sigma^2$ does not require an activation function, since it can be positive or negative. So using $\log\sigma^2$ is more convenient in the training process.

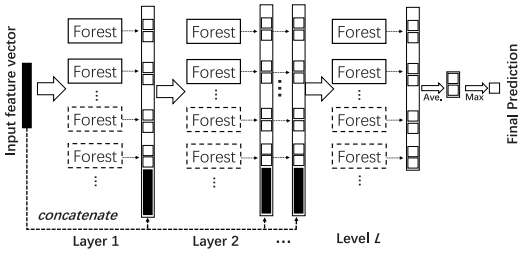


Fig. 2. Overall architecture of a DF. Forests in solid-line frames refer to completely random forests, and forests in dashed-line frames refer to random forests. Each forest outputs a two-dimensional class vector, which is then concatenated with the input feature vector to act as the input to the next layer.

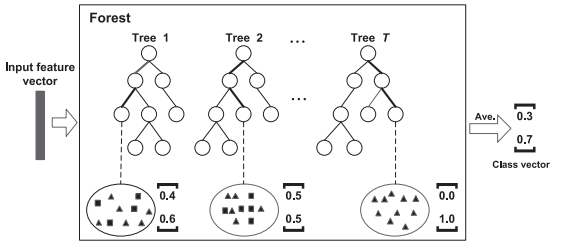


Fig. 3. Illustration of class vector generation. Different marks in leaf nodes imply different classes. Each forest will produce an estimate of the class vector by counting the percentage of different classes of training examples at the leaf node where the concerned sample falls and then averaging across all trees in the same forest.

where $P_0^{(l,m)}(x_i) = \frac{1}{T}\sum_{t=1}^{T} P_0^{(l,m,t)}(x_i)$, $P_1^{(l,m)}(x_i) = \frac{1}{T}\sum_{t=1}^{T} P_1^{(l,m,t)}(x_i)$. Specifically, to reduce the risk of overfitting, the class vector $P^{(l,m)}$ produced by each forest is generated by K-fold cross-training (Zhou & Feng, 2019). The detailed steps are as follows: (1) divide the training set into K equal sub-datasets at random; (2) randomly select $K-1$ sub-datasets to train the forest, and then use this forest to obtain the classification results for these training samples; (3) repeat the previous step K times to obtain K different classification results; and (4) take the average of these classification results. In other words, each sample is used as training data $K-1$ times, resulting in $K-1$ class vectors, which are averaged to produce the final class vector. Third, the final class vectors from all M forests are concatenated to obtain a $2M$-dimensional class vector:

$$P^{(l)}(x_i) = \left(P^{(l,1)}(x_i), P^{(l,2)}(x_i), \ldots, P^{(l,M-1)}(x_i), P^{(l,M)}(x_i)\right), \tag{8}$$

which is used as an augmented feature vector F_{aug} and concatenated with the raw feature vector F to form the new input feature F_{new} of the next layer. These M forests are merged to form layer l of the forest. Fourth, after expanding a new layer, the performance of the whole DF is estimated on the validation set. The training process terminates if there is no significant improvement in classification accuracy on the validation set. Finally, the trained DF model is used to classify the test samples. That is, all the output class vectors of the termination layer are averaged, and then the class labels of the test samples are determined according to the maximum possible criterion.

Compared with DNNs, DFs have the following advantages: (1) a self-adapting mechanism determines the number of layers, making DFs suitable for different scales of training datasets; and (2) fewer hyperparameters and less dependence on these hyperparameters, which leads to robust performance when dealing with different datasets.

3.3. VAE–DF model

To solve the class-imbalance problem of credit scoring in internet finance, we combine the advantages of the VAE and DF to construct a novel deep ensemble model, VAE–DF. The basic idea of this model is as follows. We first use the VAE to learn the feature distribution of minority class samples and generate new minority class samples. Next, we train the deep ensemble DF classifier using the

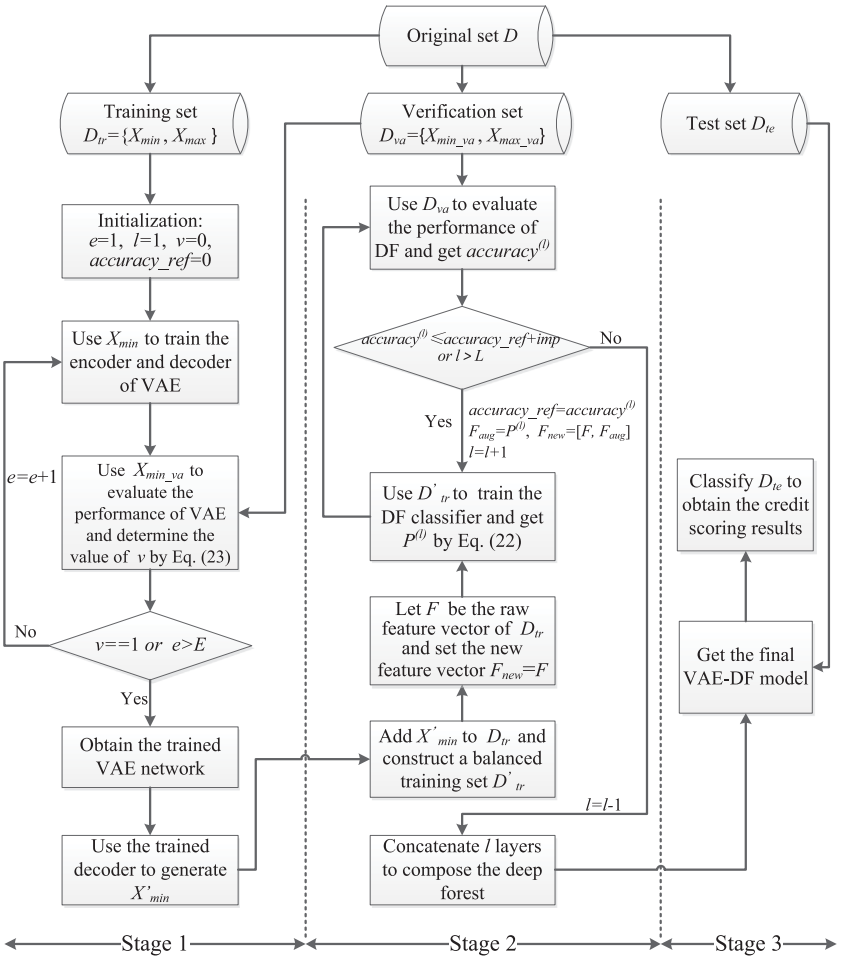


Fig. 4. Flow chart of the VAE–DF model.

balanced training set. Finally, the trained model is used to classify the test samples.

Let D be an original credit scoring dataset in internet finance. We randomly divide D into three groups: a training set, a verification set, and a test set, denoted as D_{tr}, D_{va}, and D_{te}, respectively. Let $D_{tr} = \{X_{min}, X_{maj}\}$, where X_{min} and X_{maj} respectively denote the minority and majority class samples in D_{tr}. Similarly, let $D_{va} = \{X_{min\_va}, X_{maj\_va}\}$, where X_{min_va} and X_{maj_va} respectively denote the minority and majority class samples in D_{va}. Let E be the maximum number of epochs for stopping iterations in the VAE network and L be the maximum number of layers in the DF. Let *imp* be the improvement in classification accuracy on D_{va}, which means that an improvement of less than *imp* is considered insignificant.

A flow chart of the VAE–DF model is shown in Fig. 4. There are three main stages: (1) train the VAE to balance the training set; (2) train the deep ensemble DF classifier using the balanced training set; and (3) classify the test samples. The detailed steps of the VAE–DF model are shown next.

Stage 1: Train the VAE to balance the training set

(1) Initialize the parameters: the number of epochs in the VAE network is set to $e = 1$; the number of layers in the DF is set to $l = 1$; the Boolean variable of whether to stop training the VAE network is set to $v = 0$; and the classification accuracy for reference is set to $accuracy\_ref = 0$.

(2) Use X_{min} to train the encoder and decoder of the VAE according to Fig. 1.

(3) Use X_{min_va} to evaluate the performance of the VAE network and determine the value of v according to the following equation:

$$v = \begin{cases} 1 & \text{if the validation loss no longer decreases} \\ & \text{over } pat \text{ epochs;} \\ 0 & \text{else,} \end{cases} \tag{9}$$

where *pat* is denoted as the patience periods. If $v == 1$ or $e > E$, then stop updating the weights and obtain the trained VAE network; else set $e = e + 1$ and go to Step (2) to continue training the VAE network.

(4) Use the trained decoder to generate new minority class samples X'_{min}, where the number of samples in X'_{min} is denoted as $N'_{min} = |X_{maj}| - |X_{min}|$.

Stage 2: Train the deep ensemble DF classifier using the balanced training set

(5) Add X'_{min} to D_{tr} and construct a balanced training set D'_{tr}.

(6) Let F be the raw feature vector of D_{tr} and set the new feature vector $F_{new} = F$.

(7) Employ D'_{tr} to train the DF classifier and obtain $P^{(l)}$ using Eq. (8).

(8) Utilize D_{va} to evaluate the performance of the whole DF and obtain the classification accuracy in D_{va}, namely $accuracy^{(l)}$. Specifically, if $accuracy^{(l)} \leq accuracy\_ref + imp$ or $l > L$, then stop the training process, else (i) $accuracy\_ref = accuracy^{(l)}$ and $P^{(l)} = F_{aug}$; (ii) F is merged with F_{aug} to give the new feature vector F_{new}; and (iii) $l = l + 1$ and go to Step (7) to continue training the DF classifier.

(9) Set $l = l-1$ and concatenate the l layers to compose the DF.

Stage 3: Classify the test set

(10) Get the final VAE–DF model.

(11) Use the obtained VAE–DF model to classify D_{te} and obtain the credit scoring results.

Algorithm 1 summarizes the above steps. The VAE–DF model has the following advantages. First, it applies the VAE as an over-sampling method, enabling the feature distribution of the minority class samples to be learned and improving their classification accuracy. Moreover, it adopts the deep ensemble DF approach as the classifier, which has relatively few hyperparameters and a self-adapting depth. Finally, it is based on deep learning, which is effective at dealing with high-dimensional, large-sized, and multi-source heterogeneous data. Therefore, the VAE–DF model is expected to improve the performance of imbalanced credit scoring in internet finance.

Algorithm 1: VAE–DF

Input: $D_{tr} = \{X_{min}, X_{maj}\}$, $D_{va} = \{X_{min\_va}, X_{maj\_va}\}$, D_{te}, E, L, imp.
Output: The credit scoring results.

```
Initialize: e = 1, l = 1, v = 0, accuracy_ref = 0;
Input X_min into the encoder of the VAE;
while do
    Use X_min to train the encoder and decoder of the VAE;
    Use X_min_va to evaluate the performance of the VAE network
      and determine the value of v using Eq. (9);
    if v == 1 or e > E then
        break
    else
        e = e + 1;
N'_min = |X_maj| − |X_min|;
Randomly generate N'_min z from N(0, I) and generate X'_min;
D'_tr = {D_tr, X'_min};
Let F be the raw feature vector of D_tr and set the new feature
  vector F_new = F;
while do
    Construct layer l using D'_tr and get P^(l) using Eq. (8);
    Obtain accuracy^(l);
    if accuracy^(l) ≤ accuracy_ref + imp or l > L then
        break
    else
        accuracy_ref = accuracy^(l), P^(l) = F_aug;
        F_new = [F, F_aug], l = l + 1;
l = l − 1 and concatenate l layers to compose the deep forest;
Get the final VAE–DF model and classify D_te to obtain the credit
  scoring results.
```

4. Data

Three customer credit scoring datasets[7] The first dataset contains Lending Club loan data from the first quarter of 2016, and includes 133,887 samples and 110 features. We refer to this dataset as "Lending". The second dataset, "Prosper", contains loan data for the period 2007–2012 from the US P2P loan platform, Prosper, and includes 113,937 samples and 80 features. The final dataset, "Home", was obtained from the Home Credit internet finance institution, and consisted of 30,751 samples and 718 features. As data preprocessing is a crucial step in preparing the data for training, we performed the following steps before constructing the credit scoring models.

First, we considered the class labels of the three datasets. For the Lending dataset, we treated the variable "LoanStatus" as the class label; this includes seven categories: current, fully paid, defaulted, charged, in grace period, late 16–30 days, and late 31–120 days. Note that what we are really interested in are the repaid and defaulted loans. The "current" loans that cannot be used for analysis are deleted because we do not know whether they will default in the future. To simplify the class problem and retain most of the data, we defined the "fully paid" loans as "good credit" and the remaining five categories as "bad credit". In the Prosper dataset, the variable "LoanStatus" includes 12 categories, including cancelled, charged off, completed loan, current, defaulted, and past due, among others. Similar to the Lending dataset, the "cancelled" and "current" loans are deleted because they cannot be used for analysis. Further, the "completed" loans were defined as "good credit" and the remaining categories were defined as "bad credit". For the Home dataset, all the loans have been divided into good credit and bad credit classes, so it is not necessary to handle the class labels of this dataset. In this study, the good and bad credit samples are labeled 0 and 1, respectively.

After dealing with the class labels, all three datasets were preprocessed according to the following steps: (1) deleting features—meaningless features were deleted, such as the loan ID and the customer ID, and features with a miss rate above 60% were also deleted; (2) imputing missing values—missing values for the remaining features were imputed using a mean/mode replacement for numeric/nominal features; (3) numerical conversion—Boolean features were directly encoded as 0 or 1, continuous features were standardized using min–max normalization, and multi-valued and disordered features were converted into one-hot codes; and (4) key feature selection—as our datasets are all high-dimensional, and

[7] These three datasets are available at https://www.lendingclub.com, https://www.kaggle.com/justjun0321/prosperloandata, and https://www.kaggle.com/sibmike/home-credit-blending in internet finance were used for empirical analysis.

feature selection provides an important tool that will greatly affect the classification performance (Xiao et al., 2017), we employed the new recursive feature elimination (RFE) proposed by Su et al. (2020) to select the key features.

As one of the most popular feature selection methods, RFE generates a group of candidate subsets by iteratively eliminating the least important features from the original features (Guyon et al., 2002). However, most RFE-based methods in current studies use either the overall accuracy or subset size to select the most predictive features. Then, we employed the new RFE-based approach proposed by Su et al. (2020) to determine the optimal feature subset. This approach uses the energy minimization method to find the optimal feature subset and balance the subset size and classification accuracy simultaneously, achieving the goal of dimension reduction and accurate prediction. We used RF as the base classifier for this RFE method. We also used 10-fold cross-validation, which means that a feature is being selected 0 to 10 times, as suggested by Su et al. (2020). The goodness of fit—$Fit = \xi \cdot BA + (1-\xi) \cdot \frac{SS_{final}}{SS_{all}}$—is used to evaluate each subset. Here BA denotes the balanced accuracy; SS_{final} and SS_{all} are the size of the optimal feature subset and the number of all the input features, respectively; and ξ is a constant that is set to 0.8, following Su et al. (2020). Higher values of Fit are better.

After data preprocessing, the Lending dataset has 113,553 samples and 45 features with a imbalance ratio (IR)[8] of 3.65; the Prosper dataset contains 49,723 samples 43 features with a imbalance ratio of 2.07; and the Home dataset has 30,751 samples and 36 features with a imbalance ratio of 11.14. Processing the labels reduces the number of samples, while deleting features and feature selection steps reduces the number of features, thus leading to even smaller datasets in the experiment. In order to better investigate the impact of different imbalance ratios on credit scoring models, for each of the three datasets, we kept the number of the majority class samples unchanged, and then altered the minority class samples by random under-sampling, to construct nine datasets with varying imbalance ratios (imbalance ratio = {4, 6, 8, 10, 20, 40, 60, 80, 100}), as suggested by Marqués et al. (2013). Then, a total of 30 datasets were obtained for the experiments. Table 1 displays detailed information about the credit scoring datasets, and the selected features of the three original datasets are listed in Table B.1 in Appendix B.

Table 1
Summary of the credit scoring datasets in internet finance.

Dataset	Samples	Features	Good	Bad	IR
Lending	113,553	45	89,144	24,409	3.65
Lending (1:4)	111,430	–	–	22,286	4
Lending (1:6)	104,001	–	–	14,857	6
Lending (1:8)	100,287	–	–	11,143	8
Lending (1:10)	98,058	–	–	8914	10
Lending (1:20)	93,601	–	–	4457	20
Lending (1:40)	91,373	–	–	2229	40
Lending (1:60)	90,630	–	–	1486	60
Lending (1:80)	90,258	–	–	1114	80
Lending (1:100)	90,035	–	–	891	100
Prosper	49,723	43	33,530	16,193	2.07
Prosper (1:4)	41,913	–	–	8383	4
Prosper (1:6)	39,118	–	–	5588	6
Prosper (1:8)	37,721	–	–	4191	8
Prosper (1:10)	36,886	–	–	3353	10
Prosper (1:20)	35,207	–	–	1677	20
Prosper (1:40)	34,368	–	–	838	40
Prosper (1:60)	34,089	–	–	559	60
Prosper (1:80)	33,949	–	–	419	80
Prosper (1:100)	33,865	–	–	335	100
Home	30,751	36	28,219	2532	11.14
Home (1:4)	35,274	–	–	7055	4
Home (1:6)	32,922	–	–	4703	6
Home (1:8)	31,746	–	–	3527	8
Home (1:10)	31,041	–	–	2822	10
Home (1:20)	29,630	–	–	1411	20
Home (1:40)	28,924	–	–	705	40
Home (1:60)	28,689	–	–	470	60
Home (1:80)	28,572	–	–	353	80
Home (1:100)	28,501	–	–	282	100

5. Empirical analysis

We now evaluate the classification performance of the deep ensemble VAE–DF model on 30 credit scoring datasets in internet finance. Section 5.1 details the experimental setup, and Section 5.2 introduces the model evaluation measures used to evaluate the classification performance and compare the different models. In Section 5.3, to validate the over-sampling performance of the VAE, we compare the performance of VAE–DF with several models based on four traditional over-sampling methods and two GAN-based methods. In Section 5.4, to further evaluate the credit scoring performance of the VAE–DF model, we compare it with four benchmark scoring models.

5.1. Experimental setup

The experiments used 30 credit scoring datasets in internet finance. We applied five-fold cross-validation to the datasets. Specifically, for each dataset, the data were equally split into five sub-datasets at random. One sub-dataset was used as D_{te}, another sub-dataset was used as D_{va}, and the remaining three sub-datasets were used as D_{tr}. Five-fold cross-validation was repeated 10 times and the average values were used for comparison and analysis. For each fold, all the hyperparameters were optimized on the validation set.

Table 2 lists the tuning hyperparameters for the VAE–DF model, where d denotes the number of features of the dataset. Note that the architecture of the VAE network consists of an input layer, multiple hidden layers, and an output layer, where the multiple hidden layers include the hidden layers of the encoder, layer μ, layer $\log \sigma^2$, layer z, and the hidden layers of the decoder. Following Fajardo et al. (2018), who suggested using a symmetric structure for VAE network architectures, we set the number of hidden layers to be the same in both the encoder and the decoder.

[8] We defined the imbalance ratio as the number of majority class samples divided by the number of minority class samples (Xiao, Jia et al., 2020).

Table 2
Hyperparameter settings for VAE–DF.

Hyperparameters	Value
Input layer size in VAE	d
Hidden layer size in the encoder and decoder	{(64), (128, 64)}
Layer z size	{2, 4, 6, 8}
Activation function in the input and hidden layers	*ReLU*
Output layer size in the VAE	d
Activation function in the output layer	*sigmoid*
Batch_size	128
Number of patient epochs, *pat*	50
Maximum number of epochs to stop training the VAE networks, E	1000
Number of forests in each layer, M	{2, 4, 6, 8, 10}
Number of trees in each forest, T	{400, 600, 800, 1000, 1200}
Number of candidate features used to make a split	$\sqrt{d}$
Maximum number of layers for DF, L	10
Improvement in classification accuracy for DF, *imp*	0.01
K-fold cross-training for generating the class vectors in each forest	5

Note: We adopt random forests and completely random forests in each layer, and set the number of random forests to be the same as the number of completely random forests.

Table 3
Confusion matrix for credit scoring.

	Classified positive	Classified negative
Actually positive	*TP*	*FN*
Actually negative	*FP*	*TN*

To validate the over-sampling performance of the VAE–DF model, we combined the DF with four traditional over-sampling methods and two GAN-based over-sampling methods and constructed the corresponding models. The traditional over-sampling methods include ROS, SMOTE, borderline-SMOTE (BLSMOTE), and ADASYN. As another deep generative model, the GAN has been used as an over-sampling method for class-imbalanced credit scoring problems. So we considered two types of GAN: a standard GAN method, and the cWGAN proposed by Engelmann and Lessmann (2021). The hyperparameter settings of the four SMOTE-style methods, the GAN, and the cWGAN are listed in Appendix C.1.

We also compared VAE–DF with different credit scoring models. For a fair comparison, we combined the VAE with four conventional and state-of-the-art credit scoring techniques: logistic regression (LR), a multilayer perceptron (MLP), RF, and eXtreme gradient boosting (XGBoost). Both LR and the MLP are single models. LR is a conventional credit scoring model which estimates the posterior probability, whereas MLP is a semi-parametric model and requires the modeler to select a specific prior (Lessmann et al., 2015). In addition, we adopted a wild and deep architecture for the MLP, which is different from the architecture used by Lessmann et al. (2015). RF and XGBoost are ensemble methods, both of which are state-of-the-art credit scoring models. As a result, the VAE–DF model was compared with the VAE–LR, VAE–MLP, VAE–RF, and VAE–XGBoost models. The four compared models have the same hyperparameter settings as the VAE networks in the VAE–DF model. The hyperparameter settings of these benchmark credit scoring models are given in Appendix C.2.

Over-sampling was implemented through ROS, SMOTE, borderline-SMOTE, and ADASYN based on the Python package Imbalanced-Learn 0.6.2. LR, RF, and XGboost were based on the Python package Scikit-Learn 0.22.2. The VAE and GAN networks were implemented on the TensorFlow 2.7.0 deep learning framework. The MLPs were implemented on Keras 2.7.0. All experiments were implemented on Python 3.7.3. The code for our model is available at https://github.com/sanwenyu2022/vae_df_code.

5.2. Model evaluation measures

For predictive classification in credit scoring, each loan applicant is classified into one of two classes: good credit or bad credit. In the context of imbalanced data, the minority class and majority class are generally marked as the positive class and the negative class, respectively (Nanni et al., 2015). The confusion matrix in Table 3 is commonly used as the basis for various evaluation measures in credit scoring, where *TP* (true positive) represents the number of samples that are actually positive and correctly classified as positive, *TN* (true negative) represents the number of samples that are actually negative and correctly classified as negative, *FP* (false positive) represents the number of samples that are actually negative yet misclassified as positive, and *FN* (false negative) represents the number of samples that are actually positive yet misclassified as negative. In this paper, we considered four comprehensive evaluation measures to validate the classification performance: the area under the receiving operator curve (AUC-ROC), the area under the precision recall curve (AUC-PR), the positive Brier score BS^+, and the negative Brier score BS^-.

- The AUC-ROC is the area under the ROC curve. The x-axis of this curve represents the false positive rate $FP/(FP+TN)$, and the y-axis represents the true positive rate $TP/(TP+FN)$. Larger values of the AUC-ROC indicate better model performance.
- The AUC-PR denotes the area under the precision recall curve. The x-axis represents $Recall = TP/(TP+FN)$, and *they*-axis represents $Precision = TP/(TP+FP)$. Larger values of the AUC-RR also indicate better model performance.

- BS^+ is the average quadratic loss on each minority class sample: $BS^+ = \frac{1}{N_{min}} \sum_{i=1}^{N_{min}} (y_i - p_i)^2$, where p_i denotes the predicted probability, $y_i = 1$ denotes the actual class label of the minority class sample, and N_{min} is the number of the minority class samples.
- BS^- is the average quadratic loss on each majority class sample: $BS^- = \frac{1}{N_{maj}} \sum_{i=1}^{N_{maj}} (y_i - p_i)^2$, where p_i denotes the predicted probability, $y_i = 0$ denotes the actual label of the majority class sample, and N_{maj} is the number of the majority class samples.

The AUC-ROC is widely used in credit scoring and measures a classifier's ability to discriminate between the minority class and the majority class. The AUC-PR also captures how well a classifier can discriminate the classes but emphasizes the minority class. BS^+ and BS^- are the stratified Brier score (BS) proposed by Wallace and Dahabreh (2014) for imbalanced data.[9] They quantify the average deviation between predicted probabilities and the outcomes of the minority and majority class samples respectively, and illustrate whether the predicted probabilities are well calibrated. The four metrics provide different views of classification performance and do not require setting a classification cutoff. So the observed results do not depend on the approach taken to decide on the cutoff (Engelmann & Lessmann, 2021).

5.3. Performance comparison of models combining the DF classifier with different over-sampling methods

5.3.1. Comparison of the mean rankings

We compared the eight models on 30 credit scoring datasets in internet finance using the four evaluation measures. To summarize the results, we report the mean rankings aggregated over 30 datasets of each model for each evaluation measure in Table 4. The optimal value for each evaluation measure is shown in bold. The overall mean ranking of each model is listed in the final column. For transparency, the raw numerical scores per evaluation measure for each model are shown in Tables D.1–D.4 in Appendix D.1.

From Table 4, we can draw the following conclusions: (1) VAE–DF, GAN–DF, and cWGAN–DF obtain better overall mean rankings than the other five models. Particularly, VAE–DF, GAN–DF, and cWGAN–DF have better mean rankings than the other five models in terms of all evaluation measures, except for BS^-. (2) The VAE–DF model achieves the best AUC-PR, and obtains the second-best values in terms of the AUC-ROC and BS^+ among all eight models. (3) The Pure-DF model has a better overall mean ranking than the SMOTE–DF, BLSMOTE–DF, ADASYN–DF, and ROS–DF models. (4) The ROS–DF model obtains the worst mean ranking in terms of the AUC-ROC, AUC-PR, and BS^+, but has the best mean ranking in terms of BS^-.

5.3.2. Significance test for models combining the DF classifier with different over-sampling methods

We used the non-parametric statistical test method recommended by Demsar (2006) to determine whether significant differences exist among the eight models in each evaluation measure. First, we used a Friedman test (Friedman, 1937) and an Iman–Davenport test (Iman & Davenport, 1980) to detect any significant differences among the eight models. The null hypotheses of the Friedman and Iman–Davenport tests are that the eight models have the same classification performance. With 30 datasets and eight models, the Friedman test follows a χ^2 distribution with seven degrees of freedom, and the Iman–Davenport test follows an F-distribution with seven and 7×29 degrees of freedom. In our study, the significance level is set to 0.05. So the distribution values are $\chi_7^2 = 14.1$ and $F_{(7,203)} = 2.1$. The test results are presented in the last two rows of Table 4. We can see that the test values are greater than the corresponding distribution values for each evaluation measure. Therefore, we reject the null hypotheses at a 95% confidence level and conclude that there are significant differences among the eight models for each evaluation measure.

We next proceed with pairwise comparisons to compare the performance between all pairs of the models. First, we compute $z = (R_i - R_j)/\sqrt{\frac{k(k+1)}{6N}}$, where R_i and R_j are the mean rankings of the ith and jth model, respectively, k (=8) denotes the number of models, and N (=30) is the number of datasets. Then we convert the z-values into probabilities and obtain the adjusted p-values using the Bergmann–Hommel procedure (García & Herrera, 2008). Based on the test results shown in Table 5, we can conclude the following:

(1) The VAE–DF, GAN–DF, and cWGAN–DF models perform significantly better than ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF in terms of the AUC-ROC, AUC-PR, and BS^+. This indicates that when combined with the DF classifier, the three deep generative over-sampling methods perform better than these four traditional over-sampling methods according to most evaluation measures. Furthermore, there are no significant differences among the VAE–DF, GAN–DF, and cWGAN–DF models for each evaluation measure.

(2) Pure–DF is significantly better than ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF in terms of the AUC-ROC and AUC-PR but is not significantly different from these four models in terms of BS^+ and BS^-. This shows that the Pure–DF model without a data-level balancing process yields better performance than the four models based on traditional over-sampling methods. In addition, the Pure–DF model does not differ from VAE–DF, GAN–DF, and cWGAN–DF in terms of the AUC-ROC and AUC-PR.

[9] $BS = \frac{1}{N} \sum_{i=1}^{N} (y_i - p_i)^2$, where N denotes all the samples and $y_i \in \{0, 1\}$ is the actual class label. Wallace and Dahabreh (2014) found a problem with the Brier score in the case of imbalanced datasets; calibration may be good overall but poor for the minority class. A model that predicts $p_i = 0$ for every sample will appear to be well calibrated despite its manifest uselessness. This phenomenon is analogous to the accuracy measure. So BS^+ and BS^- are more appropriate for assessing calibration in imbalanced scenarios.

Table 4
Mean rankings of the eight models across 30 datasets for each evaluation measure.

Model	AUC-ROC	AUC-PR	BS^+	BS^-	Overall
Pure–DF	4.3	3.7	5.6	2.4	4.0
ROS–DF	7.9	7.8	7.6	**1.6**	6.2
SMOTE–DF	4.6	5.1	5.7	3.7	4.8
BLSMOTE–DF	5.7	5.8	5.0	4.0	5.1
ADASYN–DF	6.3	6.4	6.0	3.4	5.5
GAN–DF	3.1	3.5	2.2	6.8	3.9
cWGAN–DF	**1.6**	1.9	**1.8**	6.9	**3.1**
VAE–DF	2.5	**1.8**	2.1	6.7	3.3
Friedman $\chi_7^2 = 14.1$	152.4	159.2	159.1	132.9	
Iman–Davenport $F_{(7,203)} = 2.1$	76.8	91.0	90.7	49.9	

Note: The Pure–DF model denotes the DF model without data-level balancing approaches.

Table 5
Full-pairwise comparison of the eight models. Bold indicates $p < 0.05$.

Measure	Model	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
AUC-ROC	Pure–DF	$\mathbf{9.9408 \cdot 10^{-10}}$	**0.0003**	**0.0006**	$\mathbf{1.3419 \cdot 10^{-5}}$	1.0000	0.6362	0.6686
	ROS–DF		0.1496	0.1145	0.6362	$\mathbf{1.3145 \cdot 10^{-13}}$	**0.0000**	$\mathbf{9.3259 \cdot 10^{-15}}$
	SMOTE–DF			1.0000	1.0000	$\mathbf{6.2887 \cdot 10^{-7}}$	$\mathbf{5.1579 \cdot 10^{-8}}$	$\mathbf{1.3632 \cdot 10^{-7}}$
	BLSMOTE–DF				1.0000	$\mathbf{1.6110 \cdot 10^{-6}}$	$\mathbf{1.3632 \cdot 10^{-7}}$	$\mathbf{3.2856 \cdot 10^{-7}}$
	ADASYN–DF					$\mathbf{1.4035 \cdot 10^{-8}}$	$\mathbf{7.5487 \cdot 10^{-10}}$	$\mathbf{2.2676 \cdot 10^{-9}}$
	GAN–DF						1.0000	1.0000
	cWGAN–DF							1.0000
AUC-PR	Pure–DF	$\mathbf{9.2863 \cdot 10^{-10}}$	**0.0056**	**0.0008**	$\mathbf{2.6477 \cdot 10^{-5}}$	1.0000	0.2206	0.2206
	ROS–DF		**0.0186**	0.0690	0.3268	$\mathbf{4.8495 \cdot 10^{-12}}$	**0.0000**	**0.0000**
	SMOTE–DF			1.0000	0.9291	**0.0003**	$\mathbf{2.7726 \cdot 10^{-7}}$	$\mathbf{2.7726 \cdot 10^{-7}}$
	BLSMOTE–DF				1.0000	$\mathbf{2.9754 \cdot 10^{-5}}$	$\mathbf{1.6954 \cdot 10^{-8}}$	$\mathbf{1.6954 \cdot 10^{-8}}$
	ADASYN–DF					$\mathbf{5.5328 \cdot 10^{-7}}$	$\mathbf{9.2256 \cdot 10^{-11}}$	$\mathbf{9.2256 \cdot 10^{-11}}$
	GAN–DF						0.8434	0.8434
	cWGAN–DF							1.0000
BS^+	Pure–DF	0.3580	1.0000	1.0000	1.0000	$\mathbf{1.6301 \cdot 10^{-8}}$	$\mathbf{1.6301 \cdot 10^{-8}}$	$\mathbf{1.6301 \cdot 10^{-8}}$
	ROS–DF		0.3580	0.2149	1.0000	$\mathbf{1.2435 \cdot 10^{-14}}$	$\mathbf{1.2435 \cdot 10^{-14}}$	$\mathbf{1.2435 \cdot 10^{-14}}$
	SMOTE–DF			1.0000	1.0000	$\mathbf{1.3905 \cdot 10^{-8}}$	$\mathbf{1.3905 \cdot 10^{-8}}$	$\mathbf{1.3905 \cdot 10^{-8}}$
	BLSMOTE–DF				1.0000	$\mathbf{1.1912 \cdot 10^{-7}}$	$\mathbf{1.1912 \cdot 10^{-7}}$	$\mathbf{1.1912 \cdot 10^{-7}}$
	ADASYN–DF					$\mathbf{3.9568 \cdot 10^{-10}}$	$\mathbf{3.9568 \cdot 10^{-10}}$	$\mathbf{3.9568 \cdot 10^{-10}}$
	GAN–DF						1.0000	1.0000
	cWGAN–DF							1.0000
BS^-	Pure–DF	1.0000	1.0000	1.0000	1.0000	$\mathbf{3.6730 \cdot 10^{-11}}$	$\mathbf{3.6730 \cdot 10^{-11}}$	$\mathbf{3.6730 \cdot 10^{-11}}$
	ROS–DF		1.0000	0.6600	1.0000	$\mathbf{7.6472 \cdot 10^{-13}}$	$\mathbf{7.9936 \cdot 10^{-13}}$	$\mathbf{7.6472 \cdot 10^{-13}}$
	SMOTE–DF			1.0000	1.0000	$\mathbf{7.1428 \cdot 10^{-9}}$	$\mathbf{7.2007 \cdot 10^{-9}}$	$\mathbf{7.1428 \cdot 10^{-9}}$
	BLSMOTE–DF				1.0000	$\mathbf{1.7169 \cdot 10^{-7}}$	$\mathbf{1.8001 \cdot 10^{-7}}$	$\mathbf{1.7169 \cdot 10^{-7}}$
	ADASYN–DF					$\mathbf{4.8211 \cdot 10^{-10}}$	$\mathbf{4.8211 \cdot 10^{-10}}$	$\mathbf{4.8211 \cdot 10^{-10}}$
	GAN–DF						1.0000	1.0000
	cWGAN–DF							1.0000

Note: In this table, **0.0000** denotes that the adjust p-value of the corresponding pairwise comparison is less than 10^{-15}.

(3) In terms of BS^+, VAE–DF, GAN–DF, and cWGAN–DF perform significantly better than Pure–DF, ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF. As can be seen in Table D.3 in Appendix D.1, the former three models obtain better BS^+ values than the other five models, especially on datasets with a severe degree of imbalance. These results indicate that the three deep generative over-sampling methods have a good recognition ability of the minority class samples.
(4) However, the results of BS^- are opposite to those of BS^+. The VAE–DF, GAN–DF, and cWGAN–DF models perform significantly worse than the other five models. The reason for this may be that, when encountering highly imbalanced datasets, Pure–DF and the four models based on traditional over-sampling methods are more focused on the calibration of the majority class samples, resulting in lower BS^- values.
(5) There are no significant differences among the ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF models in all evaluation measures, except that SMOTE–DF significantly outperforms ROS–DF in terms of the AUC–PR.

5.3.3. Analysis of the effects of dataset linearity on over-sampling methods

The above experimental results demonstrate that when combined with the DF, the VAE yields better over-sampling performance than ROS, SMOTE, BLSMOTE, and

ADASYN in most evaluations and is not significantly different from the two GAN-based over-sampling methods. The excellent over-sampling performance of the VAE might stem from dataset characteristics. Therefore, we employed the linear programming method proposed by Mangasarian (1965) to determine the linearity of all the 30 datasets, in order to determine the effects of dataset linearity on the over-sampling methods.

According to their method, we obtained the α-values of all 30 datasets, where α denotes the objective function value of linear programming. The α-values of the Lending, Lending (1:4), Lending (1:6), Lending (1:8), Lending (1:10), Lending (1:20), Lending (1:40), Lending (1:60), Lending (1:80), and Lending (1:100) datasets are $1.9043 \cdot 10^{-7}$, $1.8357 \cdot 10^{-7}$, $1.6041 \cdot 10^{-7}$, $1.5538 \cdot 10^{-7}$, $1.4712 \cdot 10^{-7}$, $1.1780 \cdot 10^{-7}$, $1.0878 \cdot 10^{-7}$, $9.9288 \cdot 10^{-8}$, $9.8409 \cdot 10^{-8}$, and $9.7444 \cdot 10^{-8}$, respectively.[10] We also find that the α-values of the other 20 datasets are 0. Mangasarian (1965) suggested that a dataset is considered to be linear when $\alpha > 0$, and non-linear when $\alpha = 0$. Thus, we conclude that the 10 Lending datasets are linearly separable, and that the other 20 datasets based on Prosper and Home are linearly inseparable.

As shown in Table 6, we obtained the mean rankings of all eight models per evaluation measure on linear and non-linear datasets. From Table 6, we can conclude the following:

(1) The overall mean rankings of VAE–DF on both linear and non-linear datasets are worse than cWGAN–DF but better than GAN–DF and the other six models. In particular, VAE–DF obtains the best mean ranking in terms of the AUC-PR among all eight models on linear and non-linear datasets. When combined with the DF classifier, the results indicate that the VAE shows good over-sampling performance on linear and non-linear datasets.
(2) On linear datasets, Pure–DF has a better overall mean ranking than ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF. This indicates that not over-sampling performs better than any traditional over-sampling methods on linear datasets when DF is used as the classifier. These results are consistent with the conclusions in Lessmann et al. (2015). In addition, we find that Pure–DF has a better overall mean ranking than ROS–DF, SMOTE–DF, BLSMOTE–DF, and ADASYN–DF on non-linear datasets.
(3) The VAE–DF, GAN–DF, and cWGAN–DF models have better overall mean rankings than the four models based on the traditional over-sampling methods on both linear and non-linear datasets. In particular, the three models' overall mean rankings on the non-linear datasets are better than those on the linear datasets, which indicates that the advantages of the three deep generative over-sampling methods are more obvious on non-linear datasets. The reason for this may be that the three deep generative over-sampling methods benefit from their strong ability to estimate the complex data distribution of the minority class samples, especially on non-linear datasets.
(4) The ROS-DF model always obtains the worst overall mean rankings on both linear and non-linear datasets. Thus, we conclude that when combined with DF, ROS is not suitable for solving the class-imbalanced credit scoring problem in internet finance.

5.3.4. Effects of the imbalance ratio on classification performance

We investigated how the imbalance ratio impacts the classification performance of the eight models. In particular, the raw scores of the eight models on the nine datasets with imbalance ratios of 4, 6, 8, 10, 20, 40, 60, 80 and 100 were used for a comparison. Fig. 5 illustrates the impact of different imbalance ratios on each evaluation measure of the eight models on the three credit scoring datasets. From Fig. 5, we can see that a higher imbalance ratio significantly impacts credit scoring performance. The detailed conclusions are as follows:

(1) We can see that the AUC-ROC values of all eight models slightly decrease as the imbalance ratio increases, indicating that a higher imbalance ratio leads to worse credit performance for these models. Moreover, as the imbalance ratio increases, the VAE–DF, GAN–DF, and cWGAN–DF models exhibit a slighter decrease in AUC-ROC values, and their advantage over the other five models increases. This result indicates that the three deep generative over-sampling methods have relatively good stability in terms of the AUC-ROC.
(2) We also find that the AUC-PR values of all eight models obviously decrease as the imbalance ratio increases. This shows that the AUC-PR is more sensitive to the imbalance ratio than the AUC-ROC. Particularly, as the imbalance ratio increases from 40 to 100, the AUC-PR values gradually decrease to zero on the two non-linearity datasets. In addition, the VAE–DF, GAN–DF, and cWGAN–DF models yield higher AUC-PR values than the other five models in most cases.
(3) From the bottom two rows, we can observe that the VAE–DF, GAN–DF, and cWGAN–DF models obtain lower BS^+ and higher BS^- values than the other five models in most cases. These results show that the three generative over-sampling methods exhibit good recognition ability of minority class samples by learning the feature distributions of the minority class samples and generating new minority samples consistent with their original distribution. The three generative over-sampling methods are more focused on calibrating the minority class samples than the majority class samples, leading to good BS^+ and bad BS^-.

[10] We used Matlab software to calculate α-values, so the numerical computation of α was done in double precision, with about 16 digits of accuracy (Sigmon & Davis, 2004). Therefore, we considered α-values less than 10^{-15} to be 0 in this paper.

Table 6
Mean rankings of eight models per evaluation measure on linear and non-linear datasets. The best ranking per measure is shown in bold.

Linearity	Measure	Pure–DF	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
Linear datasets	AUC-ROC	4.5	7.9	4.3	5.5	6.1	4.1	**2.1**	2.5
	AUC-PR	4.1	7.8	3.9	5.6	6.4	3.9	2.3	**2.0**
	BS^+	5.5	7.0	5.6	5.5	5.8	2.5	**1.9**	2.0
	BS^-	**2.3**	2.2	3.2	3.7	3.6	6.8	7.1	7.1
	Overall	4.1	6.2	4.2	5.1	5.6	4.3	**3.3**	3.4
Non-linear datasets	AUC-ROC	3.5	8.0	5.5	5.9	6.6	2.7	**1.4**	2.5
	AUC-PR	3.5	7.8	5.7	5.9	6.4	3.3	1.8	**1.7**
	BS^+	5.6	7.9	5.7	4.8	6.1	2.0	**1.8**	2.2
	BS^-	2.5	**1.3**	4.0	4.1	3.3	6.8	6.8	6.6
	Overall	3.8	6.2	5.2	5.2	5.6	3.7	**2.9**	3.2

(4) In contrast, the new minority class samples generated by the traditional over-sampling methods seem to deviate from the original distributions. This introduces external noise and leads to a classification bias towards the majority class samples. The traditional over-sampling methods are more focused on calibrating the majority class samples, and many minority class samples are misclassified as majority class samples, thus leading to bad BS^+ and good BS^-.
(5) On the linear dataset Lending, there is a slight gap in terms of BS^+ between VAE–DF and the three models based on SMOTE-style methods as the imbalance ratio increases. Particularly on Prosper and Home, which are non-linear datasets, VAE–DF, GAN–DF, and cWGAN–DF have stable BS^+ values, while the other five models' BS^+ values gradually increase to 1 as the imbalance ratio increases. The results indicate that when combined with DF, the three generative over-sampling methods exhibit good recognition ability of the minority class samples, especially on highly imbalanced and non-linear datasets.

5.4. Performance comparison of different credit scoring models

5.4.1. Comparison of the mean rankings

Table 7 presents the mean rankings aggregated over 30 datasets of the five credit scoring models for each evaluation measure. The optimal value for each evaluation measure is shown in bold. The overall ranking of each model is listed in the final column. The raw numerical scores per evaluation measure for each model are shown in Tables D.5–D.8 in Appendix D.2.

From Table 7, we can observe the following: (1) The VAE–DF model obtains the best overall mean ranking among all five models and has better mean rankings than the other four models in terms of all evaluation measures. (2) The VAE–XGBoost model obtains the second-best overall mean ranking among all five models. (3) The VAE–MLP model has a worse overall mean ranking than the VAE–RF model and yields the worst mean ranking in terms of the AUC-PR and BS^- among all five models. And (4) VAE–LR yields the worst overall mean ranking among all five models.

Table 7
Mean rankings of the five models across 30 datasets for each evaluation measure.

Model	AUC-ROC	AUC-PR	BS^+	BS^-	Overall
VAE–LR	3.7	3.9	3.6	3.7	3.7
VAE–MLP	3.6	4.1	2.7	3.8	3.5
VAE–RF	3.6	3.0	3.6	2.2	3.1
VAE–XGBoost	2.2	2.7	3.2	3.3	2.8
VAE–DF	**2.0**	**1.4**	**1.9**	**2.1**	**1.8**
Friedman $\chi^2_4 = 9.5$	34.0	57.5	24.7	32.8	
Iman-Davenport $F_{(4,116)} = 2.4$	11.5	26.7	7.5	10.9	

5.4.2. Significance test for different credit scoring models

We also used the non-parametric statistical test method recommended by Demsar (2006) to determine whether significant differences exist among the five models in each evaluation measure. With 30 datasets and five models, the Friedman test follows a χ^2 distribution with four degrees of freedom, and the Iman–Davenport test follows an F-distribution with four and 4×29 degrees of freedom. So the distribution values at the significance level of 0.05 are $\chi^2_4 = 9.5$ and $F_{(4,116)} = 2.4$. The test results are presented in the last two rows of Table 7. We can see that the test values are greater than the corresponding distribution values for each evaluation measure. Therefore, we reject the null hypotheses at a 95% confidence level and conclude that there are significant differences among the five models for each evaluation measure.

We next proceed with pairwise comparisons to compare the performance between all pairs of the models. Again, we used the Bergmann–Hommel procedure (García & Herrera, 2008) to obtain the adjusted p-values. Based on the test results shown in Table 8, we can conclude the following:

(1) In terms of the AUC-ROC, VAE–DF performs significantly better than VAE–LR, VAE–MLP, and VAE–RF but is not significantly different from VAE–XGBoost. VAE-XGBoost significantly outperforms the VAE–LR, VAE–MLP, and VAE–RF models.
(2) In terms of the AUC-PR, VAE–DF significantly outperforms the other four models. VAE–XGBoost and VAE–RF significantly outperform VAE–LR and VAE–MLP, but there is no significant difference between them.
(3) As for BS^+, VAE–DF significantly outperforms the other four models, and there are no significant

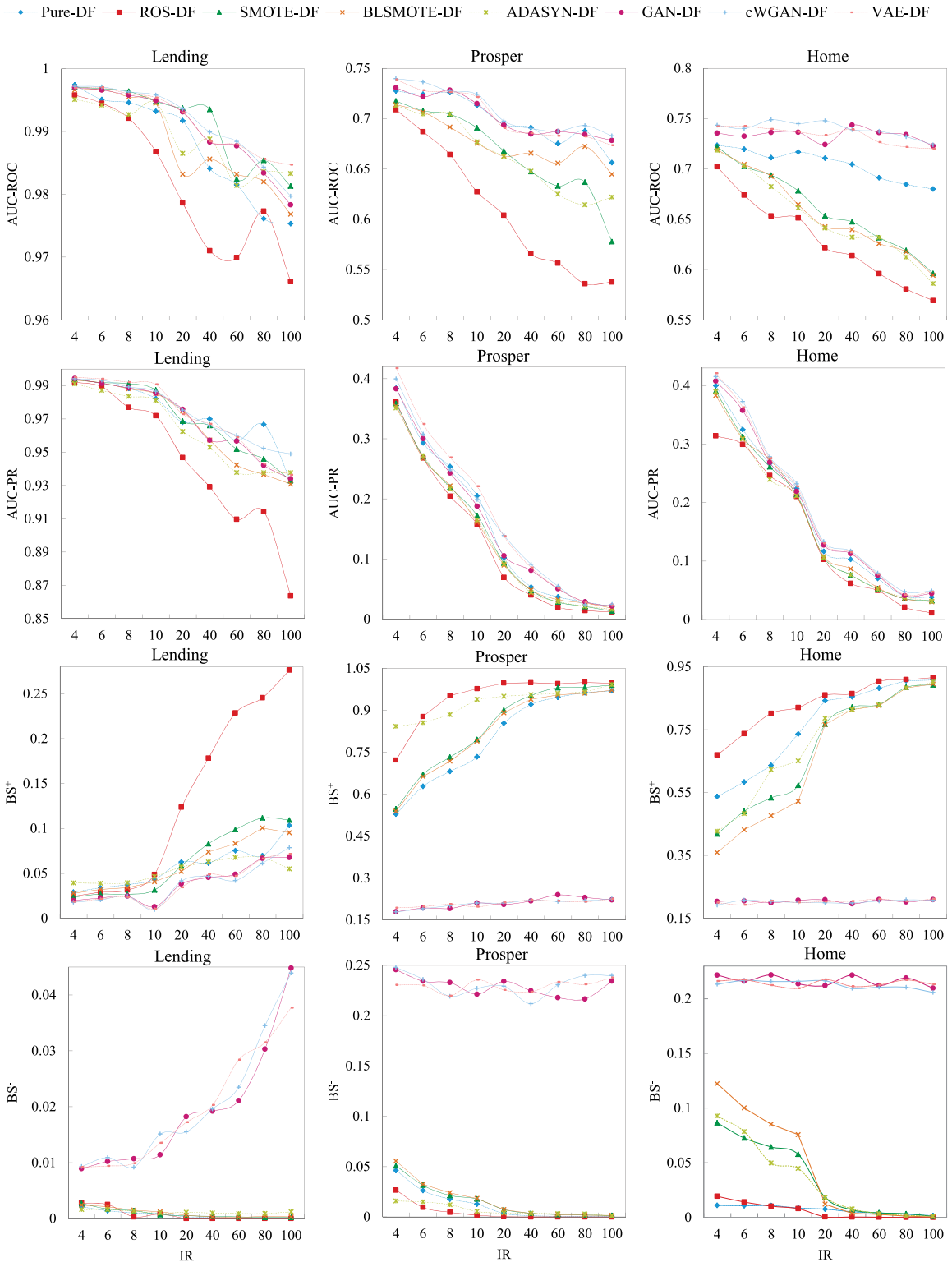


Fig. 5. Effects of the imbalance ratio on each evaluation measure.

Table 8
Full-pairwise comparison of the five models. Bold indicates $p < 0.05$.

Measure	Model	VAE–MLP	VAE–RF	VAE–XGBoost	VAE–DF
AUC-ROC	VAE–LR	0.9570	0.7437	**0.0025**	**0.0006**
	VAE–MLP		0.4369	**0.0003**	$\mathbf{4.3176 \cdot 10^{-5}}$
	VAE–RF			**0.0286**	**0.0110**
	VAE–XGBoost				0.9570
AUC-PR	VAE–LR	0.7744	**0.0002**	$\mathbf{9.3358 \cdot 10^{-5}}$	$\mathbf{1.4604 \cdot 10^{-11}}$
	VAE–MLP		**0.0027**	**0.0016**	$\mathbf{3.1094 \cdot 10^{-9}}$
	VAE–RF			0.7983	**0.0103**
	VAE–XGBoost				**0.0116**
BS^+	VAE–LR	0.7295	1.0000	0.7295	**0.0005**
	VAE–MLP		1.0000	1.0000	**0.0480**
	VAE–RF			1.0000	**0.0034**
	VAE–XGBoost				**0.0368**
BS^-	VAE–LR	**0.0237**	**0.0269**	1.0000	**0.0338**
	VAE–MLP		$\mathbf{4.7164 \cdot 10^{-7}}$	**0.0338**	$\mathbf{4.3723 \cdot 10^{-6}}$
	VAE–RF			**0.0141**	1.0000
	VAE–XGBoost				**0.0338**

differences among the other four models. The results indicate that the VAE–DF model has better recognition ability for minority class samples compared with all of the four benchmark credit scoring models.

(4) As for BS^-, VAE–DF significantly outperforms the VAE–LR, VAE–MLP, and VAE–XGBoost models but is not significantly different from the VAE-RF model. The VAE–RF model significantly outperforms the VAE–LR, VAE–MLP, and VAE–XGBoost models.

5.4.3. Feature importance analysis

Feature importance measures a feature's contribution to the prediction results for a model. The higher the feature importance value, the greater the impact on a model's prediction results. We can determine which features play an essential role in a model by ranking the feature importance. In this subsection, we propose a new method to evaluate the feature importance of the VAE–DF model. Our method is based on the permutation feature importance (Altmann et al., 2010), a model inspection technique that can be used for any fitted estimator when the data are tabular. The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled (Breiman, 2001). Compared with impurity-based feature importance, permutation feature importance has the following advantages: (1) It can overcome the defects of being strongly biased towards binary features or categorical features with a large number of possible categories. And (2) it can be applied to any black-box model and is not unique to tree-based models.

Different reference scores can be selected to calculate the permutation feature importance. We take the AUC-ROC as an example to explain how we calculate the feature importance of VAE–DF. The detailed procedure is described in Algorithm 2. First, we calculate the permutation feature importance of a single feature in each forest on the held-out validation dataset. For detailed steps for calculating the permutation feature importance, see Steps 5–10 in Algorithm 2. Next, we obtain the feature importance of each layer in the VAE–DF model by simply taking the average of the values of each layer's forests. Finally, we summ all layers' permutation feature importance values and take the average to obtain the final result. By this method, we calculated the permutation feature importance of each feature for the credit scoring dataset and then sorted all the features in descending order of the feature importance values.

Fig. 6 shows the permutation feature importance measures for the 10 most important features of the VAE–DF

Algorithm 2: Permutation feature importance of the VAE–DF model.

Input: D_{va}; number of repetitions, R; number of features, d.
Output: Permutation feature importance of VAE–DF

1 Initialize: $R = 30$;
2 $U = D_{va}$;
3 **for** *each layer* $l \in [1, L]$ **do**
4 **for** *each forest* $m \in [1, M]$ **do**
5 Compute the AUC-ROC value of the m-th forest on dataset U, and let it be $s^{l,m}$;
6 **for** *each feature* $j \in [1, d]$ **do**
7 **for** *each repetition* $r \in [1, R]$ **do**
8 Randomly shuffle feature j of dataset U to generate a corrupted version of the dataset named $\tilde{U}_{r,j}$;
9 Compute the AUC-ROC value $s^{l,m}_{r,j}$ of the m-th forest on corrupted data $\tilde{U}_{r,j}$;
10 Compute importance $FIM^{l,m}_j$ for feature j:
$$FIM^{l,m}_j = s^{l,m} - \frac{1}{R}\sum_{r=1}^{R} s^{l,m}_{r,j};$$
11 Obtain the average feature importance for feature j of layer l:
$$FIM^{l}_j = \frac{1}{M}\sum_{m=1}^{M} FIM^{l,m}_j;$$
12 Obtain the final feature importance of feature j by taking the average values of all layers:
$$FIM_j = \frac{1}{L}\sum_{l=1}^{L} FIM^{l}_j.$$

Table 9
Ten most important features based on VAE–RF.

No.	Lending	Prosper	Home
1	Total_rec_prncp	Estimated_effective_yield	New_ext_sources_mean
2	Total_pymnt_inv	Monthly_loan_payment	Ext_source_2
3	Total_pymnt	Stated_monthly_income	Ext_source_3
4	Last_pymnt_amnt	Total_inquiries	Days_birth
5	Installment	Term	Days_employed
6	Collection_recovery_fee	Loan_original_amount	New_credit_to_annuity_ratio
7	Recoveries	Estimated_return	Approved_app_credit_perc_mean
8	Loan_amnt	Inquiries_last_6months	Instal_amt_payment_min
9	Funded_amnt_inv	Debt_to_income_ratio	Instal_days_payment_std
10	Funded_amnt	Available_bankcard_credit	Prev_hour_appr_process_start_mean

model on the three original datasets, Lending, Prosper, and Home,[11] where the *y*-axis denotes the feature importance measure, and the *x*-axis denotes the specific feature names. Note that a higher feature importance value indicates a more important feature. From this figure, we can observe that some features related to income, loan amount, and repayment amount are vital to distinguishing bad credit applicants. Income is an essential financial feature that indicates the loan applicant's repaying ability (Mpofu & Mukosera, 2014). The loan amount and repayment amount will affect the behavioral scoring of the customer (Thomas, 2000). Therefore, we find that most of these features are consistent with domain knowledge.

We also obtained the 10 most important features for the VAE–RF model, as shown in Table 9. By comparison, we find that the number of the same features obtained from VAE–DF and VAE–RF is 10, 8, and 6 on the Lending, Prosper, and Home datasets, respectively. These results show that among the 10 most important features, most of the important features of the VAE–DF model are consistent with those of the VAE–RF model, which illustrates another aspect of the effectiveness of our method.

6. Conclusion

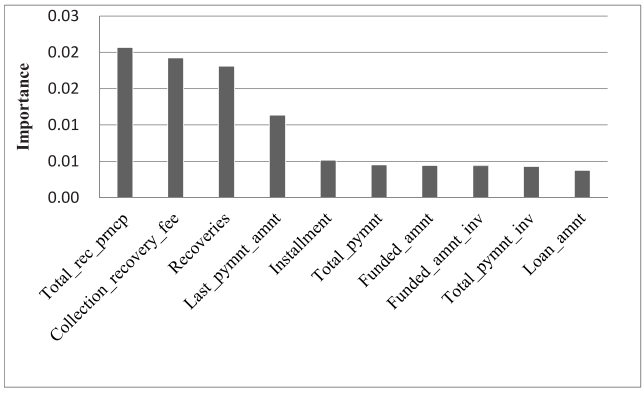
In this paper, we proposed a novel deep ensemble model for imbalanced credit scoring in internet finance. The VAE–DF model not only deals with the class-imbalance problem through VAE over-sampling but also realizes a self-adapting depth by using the deep ensemble DF classifier. The performance of the deep ensemble VAE–DF model was empirically tested using a number of internet financial credit scoring datasets in terms of four evaluation measures. We showed that the VAE–DF model performs better than the benchmark credit scoring models. We also obtained the important features that have the greatest effect on the VAE–DF model.

In addition, we applied linearity analysis to 30 credit scoring datasets. When DF was used as the classifier, the results showed that VAE over-sampling outperformed ROS and SMOTE-style methods on non-linear and linear datasets. Furthermore, we found similar conclusions for GAN–DF and cWGAN–DF by comparison with Pure–DF, ROS–DF, and three other models based on SMOTE-style methods. We attribute those results to the characteristics of the credit data in our study. Credit scoring data in internet finance may exhibit a complex structure, and our study found that traditional over-sampling methods such as ROS and SMOTE-style methods do not deal with this case. However, deep generative methods such as the VAE and GAN can cope with the challenge well by learning the complicated distributions of the non-linear data, and our results show that VAE exhibits good performance when used as an over-sampling method for the DF classifier.

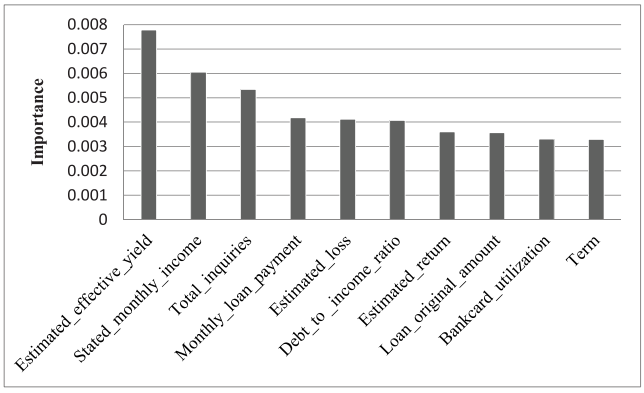
Most datasets in internet finance credit scoring are highly class-imbalanced and non-linear, so we suggest that analysts examine the imbalance ratio and the linearity of a dataset before constructing classification models. Based on the results of our study, we conclude the following:

(1) The VAE–DF model is an effective deep ensemble model, especially on highly class-imbalanced and non-linear datasets. As such, it can be applied to credit scoring tasks in internet finance institutions.
(2) VAE-based over-sampling achieves better performance than traditional over-sampling methods such as ROS, SMOTE, borderline-SMOTE, and ADASYN when using DF as the classifier.
(3) When combined with the DF classifier, deep generative over-sampling methods are a better choice for non-linear datasets. Previous studies have demonstrated that GAN-based over-sampling methods show good performance, and our study found that the VAE also exhibits good performance when used as an over-sampling method for the DF classifier.
(4) cWGAN performed very well among the over-sampling methods compared in this paper. Particularly, it ranked first in two of four evaluation measures and obtained the best overall mean ranking among all the over-sampling methods on the non-linear datasets. The results indicate that it successfully estimates the distribution of complex internet finance credit scoring datasets with numerical and categorical features and generates synthetic yet realistic data. We attribute its good performance to the auxiliary classifier (AC) loss of cWGAN, which encourages cWGAN to generate recognizable samples belonging to the conditioned class and thus is biased away from generating samples that occur in the original data but do not clearly belong to a specific class.

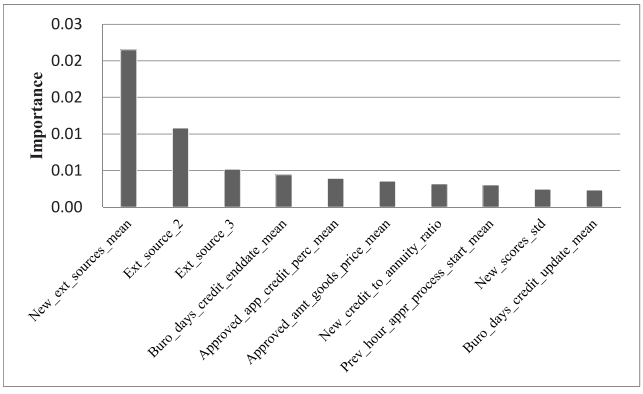
[11] We obtained a total of 30 datasets for the experiments. To conserve space, we list only the 10 most important features of the three original datasets—Lending, Prosper, and Home—with an imbalance ratio of 3.65, 2.07, and 11.14, respectively.



(a) Lending



(b) Prosper



(c) Home

Fig. 6. Diagram of the ten most important feature measures based on VAE–DF: (a) Lending, (b) Prosper, and (c) Home.

(5) On both linear and non-linear datasets, the ROS method ranked last in three of four evaluation measures and obtained the worst overall mean ranking among all the compared over-sampling methods. The results show that it is unsuitable for solving the class-imbalance problem for internet finance credit scoring.

Our work can be extended in several ways. First, as a limitation, we only focused on classification problems for credit scoring, namely differentiating good credit applicants from bad credit applicants. We will investigate how to determine the loan amount for good credit applicants in future work. Second, instead of the classification accuracy of the verification set, it is possible to use cost-sensitive external criteria to determine when to terminate

the training process. This is another improvement enabled by the DF classifier. Third, the choice of feature selection method could provide bias as to which downstream methods work best. However, we find that most studies in credit scoring perform feature selection independently from downstream methods when the number of the features is relatively large (Sun et al., 2018; Xiao, Zhou et al., 2020). Moreover, the focus of our paper was to illustrate a novel deep ensemble credit scoring model that has enough complexity. Therefore, to focus more on our model and control the complexity, we did not consider the effect of bias in this paper. In future research, we will consider simultaneously performing feature selection and building classifiers. Finally, as applicants' social networks are valuable for the modeling of internet financial credit scoring (Freedman & Jin, 2017), these networks could be utilized to build more credit scoring models based on deep ensemble learning.

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.

Acknowledgments

Financial support from the National Natural Science Foundation of China (Grant No. 72171160), the Tianfu Ten-thousand Talents Program of Sichuan Province, China (Grant No. 0082204151153) and the National Leading Talent Cultivation Project of Sichuan University, China (Grant No. SKSYL202103) is gratefully acknowledged.

Appendix A. Detailed derivation process of the VAE algorithm

A.1. Derivation of Eq. (1) in Section 3.1

$$\begin{aligned}\log p_\theta(x_i) &= \int q_\phi(z|x_i)\log p_\theta(x_i)dz \\ &= \int q_\phi(z|x_i)\log\frac{p_\theta(z,x_i)}{p_\theta(z|x_i)}dz \\ &= \int q_\phi(z|x_i)\log\frac{q_\phi(z|x_i)p_\theta(z,x_i)}{p_\theta(z|x_i)q_\phi(z|x_i)}dz \\ &= \int q_\phi(z|x_i)\log\frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}dz \\ &\quad + \int q_\phi(z|x_i)\log\frac{p_\theta(z,x_i)}{q_\phi(z|x_i)}dz \\ &= D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z|x_i)\right] + \mathcal{L}(\theta,\phi;x_i).\end{aligned} \tag{A.1}$$

A.2. Derivation of Eq. (2) in Section 3.1

$$\begin{aligned}\mathcal{L}(\theta,\phi;x_i) &= \int q_\phi(z|x_i)\log\frac{p_\theta(z,x_i)}{q_\phi(z|x_i)}dz \\ &= \int q_\phi(z|x_i)\log\frac{p_\theta(x_i|z)p_\theta(z)}{q_\phi(z|x_i)}dz \\ &= \int q_\phi(z|x_i)\log\frac{p_\theta(z)}{q_\phi(z|x_i)}dz \\ &\quad + \int q_\phi(z|x_i)\log p_\theta(x_i|z)dz \\ &= -D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right] \\ &\quad + \mathbb{E}_{q_\phi(z|x_i)}\left[\log p_\theta(x_i|z)\right].\end{aligned} \tag{A.2}$$

A.3. Derivation of Eq. (3) in Section 3.1

Suppose that $p_\theta(z) = N(z;0,I)$ and $q_\phi(z|x) = N(z;\mu,\sigma^2)$. We first deduce the case of a single normal distribution, according to the definition of KL divergence, yielding

$$\begin{aligned}D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right] &= \int \frac{1}{\sqrt{2\pi\sigma_i^2}}\exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \\ &\quad \times \log\frac{\exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)/\sqrt{2\pi\sigma_i^2}}{\exp\left(-\frac{x^2}{2}\right)/\sqrt{2\pi}}dz \\ &= \int \frac{1}{\sqrt{2\pi\sigma_i^2}}\exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \\ &\quad \times \log\left[\frac{1}{\sqrt{\sigma_i^2}}\exp\left(\frac{x^2}{2}-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)\right]dz \\ &= \frac{1}{2}\int \frac{1}{\sqrt{2\pi\sigma_i^2}}\exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) \\ &\quad \times \left(-\log\sigma_i^2 + x^2 - \frac{(x-\mu_i)^2}{\sigma_i^2}\right)dz.\end{aligned} \tag{A.3}$$

The above equation is the sum of three terms. The first term is actually an integral of $-\log\sigma_i^2$ multiplied by the probability density (whose value equals 1) and thus the result is equal to $-\log\sigma_i^2$. The second term is actually the second moment of the normal distribution, so its value is $\mu_i^2+\sigma_i^2$. For the third term, we can write it as an expectation, so it is equal to -1. Therefore, $D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right]$ is given by

$$D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right] = \frac{1}{2}\left[-\log\sigma_i^2 + \mu_i^2 + \sigma_i^2 - 1\right]. \tag{A.4}$$

Similarly, we can obtain $D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right]$ in the case of a multivariate normal distribution (see Odaibo, 2019), which can be written as

$$\begin{aligned}&D_{KL}\left[q_\phi(z|x_i)\parallel p_\theta(z)\right] \\ &= \frac{1}{2}\sum_{j=1}^{J}\left[-\log\left(\left(\sigma_i^j\right)^2\right) + \left(\mu_i^j\right)^2 + \left(\sigma_i^j\right)^2 - 1\right],\end{aligned} \tag{A.5}$$

Table B.1
Selected features of the three original credit scoring datasets in internet finance.

No.	Lending	Prosper	Home
1	Loan_amnt	Borrower_APR	Amt_annuity
2	Funded_amnt	Borrower_rate	Days_birth
3	Funded_amnt_inv	Lender_yield	Days_id_publish
4	Int_rate	Estimated_effective_yield	Days_registration
5	Installment	Estimated_loss	Ext_source_2
6	Annual_inc	Estimated_return	Ext_source_3
7	Dti	Prosper_rating	New_credit_to_annuity_ratio
8	Revol_bal	Prosper_score	New_employ_to_birth_ratio
9	Revol_util	Employment_status_duration	New_annuity_to_income_ratio
10	Total_acc	Is_borrower_homeowner	New_ext_sources_mean
11	Out_prncp	Currently_in_Group	New_scores_std
12	Out_prncp_inv	Credit_score_range_lower	New_credit_to_income_ratio
13	Total_pymnt	Credit_score_range_upper	Buro_days_credit_mean
14	Total_pymnt_inv	Current_credit_lines	Buro_days_credit_enddate_mean
15	Total_rec_prncp	Open_credit_lines	Buro_days_credit_update_mean
16	Total_rec_int	Total_credit_lines_past_7years	Buro_amt_credit_sum_mean
17	Recoveries	Open_revolving_accounts	Prev_hour_appr_process_start_mean
18	Collection_recovery_fee	Open_revolving_monthly_payment	Prev_days_decision_max
19	Last_pymnt_amnt	Inquiries_last_6months	Prev_days_decision_mean
20	Total_cur_bal	Total_inquiries	Prev_cnt_payment_sum
21	Total_rec_late_fee	Current_delinquencies	Approved_app_credit_perc_mean
22	Mths_since_rcnt_il	Amount_delinquent	Approved amt goods price mean
23	Total_bal_il	Delinquencies_last7years	Approved_days_decision_max
24	Il_util	Public_records_last10years	Instal_dbd_mean
25	Open_rv_24m	Public_records_last12months	Instal_dbd_sum
26	Max_bal_bc	Revolving_credit_balance	Instal_dbd_std
27	All_util	Bankcard_utilization	Instal_amt_instalment_max
28	Total_rev_hi_lim	Available_bankcard_credit	Instal_amt_instalment_min
29	Acc_open_past_24mths	Total_trades	Instal_amt_payment_min
30	Avg_cur_bal	Trades_never_delinquent(percentage)	Instal_amt_payment_max
31	Bc_util	Debt_to_income_ratio	Instal_amt_payment_mean
32	Mo_sin_old_il_acct	Income_verifiable	Instal_days_payment_max
33	Mo_sin_old_rev_tl_op	Stated_monthly_income	Instal_days_payment_mean
34	Mo_sin_rcnt_rev_tl_op	Loan_original_amount	Instal_days_payment_sum
35	Mths_since_recent_bc	Monthly_loan_payment	Instal_days_payment_std
36	Mths_since_recent_inq	Percent_funded	Days_employed
37	Num_rev_accts	Recommendations	
38	Num_rev_tl_bal_gt_0	Investment_from_friends_count	
39	Tot_hi_cred_lim	Investment_from_friends_amount	
40	Total_bal_ex_mort	Investors	
41	Total_bc_limit	Trades_opened_last_6months	
42	Total_il_high_credit_limit	Diff_month	
43	Grade	Term	
44	Debt_settlement_flag		
45	Total_rec_late_free		

where μ_i^j and σ_i^j denote the jth elements of μ_i and σ_i, respectively, and J denotes the dimensionality of z.

Appendix B. Selected features on the credit scoring datasets in internet finance

Table B.1 lists all the selected features of the three original credit scoring datasets, including Lending, Prosper, and Home.

Appendix C. Hyperparameter settings

C.1. Benchmark over-sampling methods

For the three SMOTE-style methods, the number of neighboring samples $k_{neighbors}$ is a hyperparameter; and for BLSMOTE, the $m_{neighbors}$ to consider for the identification of borderline samples is an additional hyperparameter. For the VAE, we tested eight candidate hyperparameter settings and thus we considered the same number for the SMOTE-style methods. For all SMOTE-style methods, we selected $k_{neighbors}$ from $\{1, 3, 5, 7, 9, 15, 20, 25\}$, except for BLSMOTE, where we selected $k_{neighbors}$ from $\{3, 5, 7, 9\}$ and $m_{neighbors}$ from $\{10, 15\}$.

For the standard GAN over-sampling method, we also tested eight candidate hyperparameter settings. We selected *batch_size* from $\{64, 128\}$ and *epochs* from $\{800, 1000\}$. The generator and discriminator layer sizes were chosen from $\{(128), (256, 128)\}$. In addition, we adopted *ReLU* as the activation function in hidden layers.

For the cWGAN method, eight candidate hyperparameter settings were tested in Engelmann and Lessmann (2021), so we employed the same settings in our paper. They made the cWGAN source code available on GitHub and provided details regarding the hyperparameter settings. Owing to space limitations, we do not reintroduce the cWGAN hyperparameter settings here. Detailed

Table D.1
AUC-ROC results in numerical format for the eight models combining the DF classifier with different over-sampling methods.

Dataset	Pure–DF	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
Lending	0.9968	0.9960	0.9970	0.9978	0.9962	0.9972	0.9976	0.9982
Lending (1:4)	0.9974	0.9958	0.9971	0.9966	0.9951	0.9970	0.9972	0.9969
Lending (1:6)	0.9951	0.9945	0.9968	0.9967	0.9942	0.9966	0.9971	0.9970
Lending (1:8)	0.9946	0.9921	0.9964	0.9955	0.9927	0.9958	0.9963	0.9960
Lending (1:10)	0.9932	0.9868	0.9948	0.9945	0.9946	0.9949	0.9958	0.9954
Lending (1:20)	0.9917	0.9786	0.9937	0.9832	0.9865	0.9931	0.9936	0.9930
Lending (1:40)	0.9841	0.9710	0.9935	0.9856	0.9888	0.9883	0.9899	0.9884
Lending (1:60)	0.9815	0.9699	0.9824	0.9832	0.9814	0.9877	0.9884	0.9879
Lending (1:80)	0.9761	0.9773	0.9854	0.9820	0.9836	0.9834	0.9843	0.9856
Lending (1:100)	0.9753	0.9661	0.9813	0.9768	0.9833	0.9783	0.9797	0.9847
Prosper	0.7364	0.7234	0.7295	0.7281	0.7243	0.7316	0.7442	0.7337
Prosper (1:4)	0.7275	0.7088	0.7174	0.7138	0.7129	0.7305	0.7396	0.7388
Prosper (1:6)	0.7241	0.6869	0.7078	0.7066	0.7044	0.7217	0.7363	0.7280
Prosper (1:8)	0.7258	0.6642	0.7043	0.6915	0.7045	0.7281	0.7271	0.7261
Prosper (1:10)	0.7132	0.6272	0.6908	0.6755	0.6768	0.7148	0.7240	0.7213
Prosper (1:20)	0.6942	0.6038	0.6678	0.6627	0.6623	0.6935	0.6975	0.6906
Prosper (1:40)	0.6911	0.5657	0.6473	0.6656	0.6477	0.6846	0.6902	0.6854
Prosper (1:60)	0.6751	0.5563	0.6331	0.6557	0.6248	0.6870	0.6874	0.6828
Prosper (1:80)	0.6877	0.5359	0.6368	0.6723	0.6142	0.6841	0.6931	0.6822
Prosper (1:100)	0.6562	0.5376	0.5777	0.6447	0.6219	0.6783	0.6828	0.6734
Home	0.7212	0.6141	0.6714	0.6636	0.6512	0.7319	0.7396	0.7445
Home (1:4)	0.7234	0.7021	0.7219	0.7190	0.7184	0.7355	0.7434	0.7422
Home (1:6)	0.7196	0.6739	0.7027	0.7045	0.7032	0.7324	0.7401	0.7425
Home (1:8)	0.7112	0.6532	0.6937	0.6925	0.6824	0.7362	0.7488	0.7396
Home (1:10)	0.7168	0.6512	0.6783	0.6644	0.6612	0.7364	0.7450	0.7368
Home (1:20)	0.7107	0.6214	0.6533	0.6425	0.6412	0.7241	0.7478	0.7335
Home (1:40)	0.7045	0.6137	0.6474	0.6395	0.6320	0.7435	0.7390	0.7392
Home (1:60)	0.6912	0.5959	0.6315	0.6258	0.6318	0.7360	0.7374	0.7265
Home (1:80)	0.6846	0.5806	0.6189	0.6175	0.6123	0.7340	0.7319	0.7218
Home (1:100)	0.6800	0.5692	0.5961	0.5942	0.5860	0.7228	0.7240	0.7200

information can be found in Engelmann and Lessmann (2021).

C.2. Benchmark credit scoring models

The VAE–LR, VAE–MLP, VAE–RF, and VAE–XGBoost models have the same hyperparameter settings as the VAE networks in the VAE–DF model.

For LR, the penalty type was set to be L2 and the inverse penalty coefficient C was varied from 5–20 in step of 5. The MLP was trained on Keras 2.7.0 and the hyperparameter settings were more complex. First, we varied the number of hidden layers from 1–10, and the number of neurons in each hidden layer was selected from $\{64, 128, 256\}$. In addition, the input layer matched the input space with d features and the output layer matched the binary output. Second, we used the *ReLU* activation function for the input layer and hidden layers, and applied the *sigmoid* function in the output layer (Srivastava et al., 2014). *ReLU* yields equal or better performance than *tanh*, and can create sparse representations with true zeros, which is remarkably suitable for naturally sparse data (Glorot et al., 2011). It is usually chosen as the activation function in deep learning models. Third, we used an input dropout ratio of 0.1 and a hidden dropout ratio of 0.25 or 0.5, according to the size of the dataset (Srivastava et al., 2014), to reduce the risk of overfitting and obtain better generalization. Finally, we employed early stopping to further reduce the risk of overfitting. Once the overall accuracy in the validation set did not decrease for *pat* = 50 periods, the training process was terminated.

For RF, we varied the number of trees in the RF according to $T \in \{200, 300, 400, 500\}$ and set the number of candidate features used to make a split $w = \sqrt{d}$. For XGBoost, the learning rate was 0.1 and the maximum tree depth was 3. Decision trees were chosen as the base estimators, and the number of estimators was chosen from $\{200, 300, 400, 500\}$.

Appendix D. Raw scores

D.1. Raw scores of the eight models combining the DF classifier with different over-sampling methods

Tables D.1–D.4 respectively report the AUC-ROC, AUC-PR, BS^+, and BS^- results in numerical format per dataset for the eight models combining the DF classifier with different over-sampling methods. For AUC-ROC and AUC-PR, higher values are better, and for BS^+ and BS^-, lower values are better.

D.2. Raw scores of the five different credit scoring models

Tables D.5–D.8 respectively report the AUC-ROC, AUC-PR, BS^+, and BS^- results in numerical format per dataset for the five different credit scoring models. For AUC-ROC and AUC-PR, higher values are better, and for BS^+ and BS^-, lower values are better.

Table D.2
AUC-PR results in numerical format for the eight models combining the DF classifier with different over-sampling methods.

Dataset	Pure–DF	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
Lending	0.9941	0.9935	0.9939	0.9938	0.9940	0.9945	0.9948	0.9955
Lending (1:4)	0.9938	0.9921	0.9936	0.9933	0.9911	0.9943	0.9941	0.9950
Lending (1:6)	0.9914	0.9893	0.9918	0.9916	0.9871	0.9913	0.9932	0.9940
Lending (1:8)	0.9883	0.9769	0.9912	0.9881	0.9835	0.9886	0.9892	0.9920
Lending (1:10)	0.9822	0.9719	0.9873	0.9851	0.9811	0.9854	0.9866	0.9907
Lending (1:20)	0.9680	0.9467	0.9686	0.9747	0.9624	0.9757	0.9753	0.9728
Lending (1:40)	0.9699	0.9291	0.9660	0.9573	0.9529	0.9572	0.9658	0.9669
Lending (1:60)	0.9570	0.9096	0.9518	0.9423	0.9378	0.9566	0.9601	0.9589
Lending (1:80)	0.9666	0.9144	0.9459	0.9366	0.9377	0.9421	0.9523	0.9430
Lending (1:100)	0.9326	0.8636	0.9328	0.9308	0.9377	0.9340	0.9489	0.9362
Prosper	0.5552	0.4890	0.5492	0.5283	0.5330	0.5428	0.5448	0.5560
Prosper (1:4)	0.3840	0.3606	0.3586	0.3544	0.3512	0.3831	0.3996	0.4173
Prosper (1:6)	0.2931	0.2680	0.2696	0.2671	0.2719	0.3003	0.3077	0.3245
Prosper (1:8)	0.2537	0.2045	0.2196	0.2216	0.2167	0.2428	0.2481	0.2689
Prosper (1:10)	0.2049	0.1575	0.1723	0.1612	0.1649	0.1878	0.1987	0.2209
Prosper (1:20)	0.1018	0.0695	0.0931	0.0900	0.0918	0.1054	0.1389	0.1377
Prosper (1:40)	0.0536	0.0402	0.0479	0.0483	0.0441	0.0812	0.0912	0.0857
Prosper (1:60)	0.0370	0.0197	0.0283	0.0325	0.0294	0.0503	0.0547	0.0512
Prosper (1:80)	0.0285	0.0143	0.0211	0.0268	0.0216	0.0288	0.0238	0.0280
Prosper (1:100)	0.0210	0.0119	0.0132	0.0183	0.0162	0.0215	0.0243	0.0223
Home	0.2191	0.1401	0.1463	0.1451	0.1417	0.2102	0.2229	0.2323
Home (1:4)	0.3997	0.3138	0.3905	0.3828	0.3894	0.4074	0.4152	0.4208
Home (1:6)	0.3248	0.2993	0.3128	0.3089	0.3088	0.3572	0.3727	0.3627
Home (1:8)	0.2666	0.2462	0.2610	0.2735	0.2393	0.2686	0.2780	0.2755
Home (1:10)	0.2238	0.2102	0.2133	0.2107	0.2115	0.2189	0.2319	0.2274
Home (1:20)	0.1162	0.1029	0.1062	0.1079	0.1058	0.1272	0.1332	0.1300
Home (1:40)	0.1028	0.0618	0.0763	0.0867	0.0764	0.1128	0.1169	0.1148
Home (1:60)	0.0701	0.0497	0.0523	0.0545	0.0516	0.0755	0.0793	0.0772
Home (1:80)	0.0399	0.0211	0.0356	0.0354	0.0385	0.0410	0.0475	0.0425
Home (1:100)	0.0384	0.0112	0.0322	0.0304	0.0311	0.0445	0.0489	0.0472

Table D.3
BS^{+} results in numerical format for the eight models combining the DF classifier with different over-sampling methods.

Dataset	Pure–DF	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
Lending	0.0201	0.0256	0.0238	0.0234	0.0278	0.0176	0.0167	0.0165
Lending (1:4)	0.0291	0.0245	0.0236	0.0279	0.0394	0.0198	0.0178	0.0184
Lending (1:6)	0.0340	0.0290	0.0270	0.0323	0.0390	0.0227	0.0205	0.0216
Lending (1:8)	0.0375	0.0315	0.0267	0.0342	0.0395	0.0246	0.0247	0.0235
Lending (1:10)	0.0433	0.0488	0.0315	0.0408	0.0464	0.0128	0.0096	0.0106
Lending (1:20)	0.0627	0.1236	0.0585	0.0521	0.0570	0.0382	0.0416	0.0345
Lending (1:40)	0.0616	0.1782	0.0830	0.0738	0.0626	0.0454	0.0468	0.0489
Lending (1:60)	0.0753	0.2286	0.0988	0.0832	0.0676	0.0490	0.0421	0.0472
Lending (1:80)	0.0697	0.2454	0.1117	0.1004	0.0676	0.0667	0.0614	0.0665
Lending (1:100)	0.1034	0.2761	0.1093	0.0952	0.0550	0.0676	0.0786	0.0714
Prosper	0.3622	0.3502	0.3471	0.3487	0.7933	0.1559	0.1701	0.1555
Prosper (1:4)	0.5282	0.7218	0.5477	0.5390	0.8426	0.1790	0.1783	0.1931
Prosper (1:6)	0.6279	0.8777	0.6709	0.6629	0.8555	0.1924	0.1900	0.1975
Prosper (1:8)	0.6814	0.9531	0.7323	0.7180	0.8845	0.1910	0.2012	0.2060
Prosper (1:10)	0.7335	0.9770	0.7952	0.7896	0.9391	0.2105	0.2102	0.1970
Prosper (1:20)	0.8537	0.9972	0.9011	0.8912	0.9501	0.2051	0.2105	0.2122
Prosper (1:40)	0.9204	0.9988	0.9529	0.9392	0.9558	0.2179	0.2210	0.2211
Prosper (1:60)	0.9465	0.9954	0.9808	0.9551	0.9603	0.2396	0.2162	0.2175
Prosper (1:80)	0.9629	0.9997	0.9828	0.9614	0.9657	0.2303	0.2188	0.2150
Prosper (1:100)	0.9697	0.9970	0.9901	0.9718	0.9917	0.2216	0.2252	0.2251
Home	0.7000	0.8304	0.7464	0.7634	0.6615	0.2055	0.1992	0.1998
Home (1:4)	0.5367	0.6693	0.4179	0.3593	0.4270	0.2036	0.1923	0.1998
Home (1:6)	0.5835	0.7370	0.4902	0.4313	0.4832	0.2054	0.2078	0.1926
Home (1:8)	0.6361	0.8017	0.5334	0.4762	0.6228	0.1991	0.2037	0.2057
Home (1:10)	0.7363	0.8203	0.5728	0.5221	0.6502	0.2072	0.2008	0.2003
Home (1:20)	0.8425	0.8600	0.7675	0.7662	0.7861	0.2089	0.1992	0.2032
Home (1:40)	0.8545	0.8645	0.8219	0.8139	0.8138	0.1959	0.1986	0.2047
Home (1:60)	0.8819	0.9038	0.8298	0.8256	0.8291	0.2096	0.2051	0.2118
Home (1:80)	0.9056	0.9090	0.8845	0.8827	0.8825	0.2021	0.2089	0.2043
Home (1:100)	0.9078	0.9158	0.8924	0.8947	0.9012	0.2098	0.2077	0.2119

Table D.4
BS^{-} results in numerical format for the eight models combining the DF classifier with different over-sampling methods.

Dataset	Pure–DF	ROS–DF	SMOTE–DF	BLSMOTE–DF	ADASYN–DF	GAN–DF	cWGAN–DF	VAE–DF
Lending	0.0032	0.0031	0.0028	0.0026	0.0018	0.0090	0.0085	0.0097
Lending (1:4)	0.0021	0.0028	0.0027	0.0024	0.0016	0.0089	0.0093	0.0092
Lending (1:6)	0.0014	0.0025	0.0017	0.0021	0.0016	0.0102	0.0109	0.0094
Lending (1:8)	0.0010	0.0003	0.0014	0.0015	0.0013	0.0107	0.0092	0.0099
Lending (1:10)	0.0009	0.0008	0.0007	0.0012	0.0010	0.0114	0.0151	0.0135
Lending (1:20)	0.0003	0.0000	0.0005	0.0006	0.0011	0.0182	0.0155	0.0172
Lending (1:40)	0.0002	0.0000	0.0004	0.0003	0.0010	0.0192	0.0196	0.0203
Lending (1:60)	0.0001	0.0000	0.0003	0.0002	0.0009	0.0211	0.0235	0.0284
Lending (1:80)	0.0001	0.0000	0.0002	0.0004	0.0009	0.0303	0.0345	0.0315
Lending (1:100)	0.0001	0.0000	0.0002	0.0003	0.0012	0.0448	0.0439	0.0377
Prosper	0.1029	0.1089	0.1125	0.1173	0.1019	0.2691	0.2536	0.2685
Prosper (1:4)	0.0462	0.0266	0.0508	0.0555	0.0158	0.2456	0.2479	0.2304
Prosper (1:6)	0.0262	0.0094	0.0314	0.0324	0.0149	0.2342	0.2361	0.2299
Prosper (1:8)	0.0177	0.0046	0.0213	0.0239	0.0121	0.2328	0.2192	0.2197
Prosper (1:10)	0.0127	0.0017	0.0178	0.0183	0.0053	0.2212	0.2272	0.2355
Prosper (1:20)	0.0034	0.0002	0.0073	0.0072	0.0038	0.2341	0.2292	0.2255
Prosper (1:40)	0.0012	0.0000	0.0036	0.0034	0.0035	0.2247	0.2118	0.2237
Prosper (1:60)	0.0006	0.0000	0.0023	0.0024	0.0032	0.2179	0.2306	0.2334
Prosper (1:80)	0.0003	0.0000	0.0018	0.0017	0.0029	0.2165	0.2398	0.2309
Prosper (1:100)	0.0002	0.0000	0.0014	0.0016	0.0009	0.2342	0.2398	0.2377
Home	0.0094	0.0075	0.0508	0.7020	0.0402	0.2162	0.2142	0.2058
Home (1:4)	0.0110	0.0193	0.0865	0.1223	0.0927	0.2216	0.2132	0.2161
Home (1:6)	0.0107	0.0141	0.0726	0.1001	0.0785	0.2161	0.2167	0.2173
Home (1:8)	0.0108	0.0103	0.0644	0.0852	0.0497	0.2217	0.2157	0.2123
Home (1:10)	0.0086	0.0080	0.0577	0.0757	0.0447	0.2136	0.2159	0.2095
Home (1:20)	0.0076	0.0005	0.0182	0.0121	0.0183	0.2120	0.2163	0.2175
Home (1:40)	0.0052	0.0004	0.0069	0.0038	0.0078	0.2216	0.2094	0.2112
Home (1:60)	0.0041	0.0002	0.0042	0.0028	0.0035	0.2121	0.2105	0.2126
Home (1:80)	0.0029	0.0000	0.0034	0.0012	0.0018	0.2189	0.2104	0.2171
Home (1:100)	0.0016	0.0000	0.0009	0.0005	0.0012	0.2096	0.2057	0.2130

Table D.5
AUC-ROC results in numerical format for the five different credit scoring models.

Dataset	VAE–LR	VAE–MLP	VAE–RF	VAE–XGBoost	VAE–DF
Lending	0.9852	0.9980	0.9965	0.9968	0.9982
Lending (1:4)	0.9802	0.9979	0.9973	0.9968	0.9969
Lending (1:6)	0.9835	0.9969	0.9962	0.9921	0.9970
Lending (1:8)	0.9864	0.9950	0.9958	0.9966	0.9960
Lending (1:10)	0.9850	0.9915	0.9941	0.9968	0.9954
Lending (1:20)	0.9829	0.9947	0.9925	0.9953	0.9930
Lending (1:40)	0.9833	0.9906	0.9834	0.9931	0.9884
Lending (1:60)	0.9827	0.9889	0.9770	0.9802	0.9879
Lending (1:80)	0.9797	0.9827	0.9785	0.9836	0.9856
Lending (1:100)	0.9713	0.9702	0.9771	0.9906	0.9847
Prosper	0.7189	0.7015	0.7387	0.7423	0.7337
Prosper (1:4)	0.7162	0.6851	0.7321	0.7363	0.7388
Prosper (1:6)	0.7121	0.7291	0.6914	0.7349	0.7280
Prosper (1:8)	0.7130	0.6618	0.7193	0.7309	0.7261
Prosper (1:10)	0.7098	0.6601	0.7127	0.7199	0.7213
Prosper (1:20)	0.7029	0.6478	0.6883	0.7084	0.6906
Prosper (1:40)	0.7033	0.6389	0.6760	0.6935	0.6854
Prosper (1:60)	0.6954	0.6495	0.6894	0.7052	0.6828
Prosper (1:80)	0.7028	0.6821	0.6737	0.6869	0.6822
Prosper (1:100)	0.6977	0.6324	0.6732	0.6850	0.6734
Home	0.7267	0.7064	0.7387	0.7331	0.7445
Home (1:4)	0.7264	0.7550	0.7240	0.7294	0.7422
Home (1:6)	0.7410	0.7420	0.7353	0.7413	0.7425
Home (1:8)	0.7340	0.7308	0.7338	0.7379	0.7396
Home (1:10)	0.7304	0.7295	0.7395	0.7350	0.7368
Home (1:20)	0.7327	0.7173	0.7127	0.7273	0.7335
Home (1:40)	0.7077	0.6699	0.7211	0.7275	0.7392
Home (1:60)	0.7059	0.6629	0.7169	0.7173	0.7265
Home (1:80)	0.6954	0.7049	0.6940	0.7009	0.7218
Home (1:100)	0.7066	0.7230	0.7081	0.6862	0.7200

Table D.6
AUC-PR results in numerical format for the five different credit scoring models.

Dataset	VAE–LR	VAE–MLP	VAE–RF	VAE–XGBoost	VAE–DF
Lending	0.9703	0.9924	0.9945	0.9913	0.9955
Lending (1:4)	0.9646	0.9943	0.9947	0.9923	0.9950
Lending (1:6)	0.9624	0.9918	0.9922	0.9903	0.9940
Lending (1:8)	0.9660	0.9902	0.9889	0.9876	0.9920
Lending (1:10)	0.9505	0.9865	0.9866	0.9858	0.9907
Lending (1:20)	0.9317	0.9745	0.9705	0.9698	0.9728
Lending (1:40)	0.8969	0.9361	0.9441	0.9528	0.9669
Lending (1:60)	0.8889	0.9402	0.9307	0.9379	0.9589
Lending (1:80)	0.8648	0.9279	0.9046	0.9090	0.9430
Lending (1:100)	0.8001	0.8832	0.9017	0.9157	0.9362
Prosper	0.5275	0.5024	0.5550	0.5649	0.5560
Prosper (1:4)	0.3712	0.3322	0.3889	0.4031	0.4173
Prosper (1:6)	0.2766	0.2259	0.3040	0.3119	0.3245
Prosper (1:8)	0.2248	0.1848	0.2391	0.2486	0.2689
Prosper (1:10)	0.1901	0.1746	0.1984	0.1988	0.2209
Prosper (1:20)	0.1063	0.0829	0.1009	0.1118	0.1377
Prosper (1:40)	0.0570	0.0422	0.0497	0.0520	0.0857
Prosper (1:60)	0.0408	0.0313	0.0379	0.0385	0.0512
Prosper (1:80)	0.0389	0.0271	0.0276	0.0299	0.0280
Prosper (1:100)	0.0282	0.0167	0.0215	0.0239	0.0223
Home	0.2065	0.1730	0.2128	0.2140	0.2323
Home (1:4)	0.4045	0.4409	0.4028	0.4520	0.4208
Home (1:6)	0.3260	0.3290	0.3724	0.3461	0.3627
Home (1:8)	0.2713	0.2433	0.2510	0.2554	0.2755
Home (1:10)	0.2187	0.2066	0.2196	0.2074	0.2274
Home (1:20)	0.1193	0.1038	0.1489	0.1678	0.1300
Home (1:40)	0.0617	0.0475	0.1107	0.0710	0.1148
Home (1:60)	0.0485	0.0303	0.0624	0.0540	0.0772
Home (1:80)	0.0310	0.0267	0.0385	0.0360	0.0425
Home (1:100)	0.0270	0.0307	0.0341	0.0350	0.0472

Table D.7
BS^{+} results in numerical format for the five different credit scoring models.

Dataset	VAE–LR	VAE–MLP	VAE–RF	VAE–XGBoost	VAE–DF
Lending	0.0522	0.0208	0.0262	0.0315	0.0165
Lending (1:4)	0.0466	0.0172	0.0271	0.0315	0.0184
Lending (1:6)	0.0301	0.0189	0.0332	0.0280	0.0216
Lending (1:8)	0.0476	0.0186	0.0353	0.0315	0.0235
Lending (1:10)	0.0510	0.0236	0.0353	0.0305	0.0106
Lending (1:20)	0.0537	0.0343	0.0427	0.0368	0.0345
Lending (1:40)	0.0473	0.0361	0.0517	0.0524	0.0489
Lending (1:60)	0.0638	0.0407	0.0588	0.0462	0.0472
Lending (1:80)	0.0641	0.0359	0.0678	0.0538	0.0665
Lending (1:100)	0.0748	0.0394	0.0754	0.0531	0.0714
Prosper	0.2080	0.2239	0.1836	0.1962	0.1555
Prosper (1:4)	0.2093	0.2275	0.1937	0.1982	0.1931
Prosper (1:6)	0.2118	0.2022	0.2198	0.2168	0.1975
Prosper (1:8)	0.2148	0.2014	0.2115	0.2109	0.2060
Prosper (1:10)	0.2153	0.2374	0.2021	0.2062	0.1970
Prosper (1:20)	0.2093	0.2655	0.2137	0.2131	0.2122
Prosper (1:40)	0.2139	0.2725	0.2129	0.2244	0.2211
Prosper (1:60)	0.2122	0.2802	0.2272	0.2222	0.2175
Prosper (1:80)	0.2236	0.2195	0.2293	0.2186	0.2150
Prosper (1:100)	0.2091	0.3044	0.2208	0.2329	0.2251
Home	0.2147	0.2014	0.2022	0.2079	0.1998
Home (1:4)	0.2157	0.1561	0.2192	0.1972	0.1998
Home (1:6)	0.2097	0.1945	0.1997	0.2044	0.1926
Home (1:8)	0.2100	0.2126	0.1993	0.1709	0.2057
Home (1:10)	0.2138	0.1939	0.2117	0.2017	0.2003
Home (1:20)	0.2143	0.2009	0.2222	0.2187	0.2032
Home (1:40)	0.2193	0.2732	0.2124	0.2254	0.2047
Home (1:60)	0.2162	0.2441	0.2387	0.2215	0.2118
Home (1:80)	0.2228	0.2795	0.2483	0.2466	0.2043
Home (1:100)	0.2133	0.1985	0.2248	0.2324	0.2119

Table D.8
BS^{-} results in numerical format per dataset for the five different credit scoring models.

Dataset	VAE–LR	VAE–MLP	VAE–RF	VAE–XGBoost	VAE–DF
Lending	0.0193	0.0073	0.0107	0.0098	0.0097
Lending (1:4)	0.0267	0.0106	0.0109	0.0090	0.0092
Lending (1:6)	0.0230	0.0096	0.0116	0.0106	0.0094
Lending (1:8)	0.0184	0.0101	0.0131	0.0128	0.0099
Lending (1:10)	0.0260	0.0152	0.0140	0.0145	0.0135
Lending (1:20)	0.0272	0.0213	0.0177	0.0167	0.0172
Lending (1:40)	0.0324	0.0218	0.0236	0.0223	0.0203
Lending (1:60)	0.0330	0.0244	0.0274	0.0297	0.0284
Lending (1:80)	0.0361	0.0253	0.0332	0.0341	0.0315
Lending (1:100)	0.0422	0.0317	0.0380	0.0384	0.0377
Prosper	0.2288	0.2341	0.2136	0.2308	0.2685
Prosper (1:4)	0.2392	0.2486	0.2244	0.2355	0.2304
Prosper (1:6)	0.2313	0.2811	0.2220	0.2397	0.2299
Prosper (1:8)	0.2259	0.2640	0.2186	0.2212	0.2197
Prosper (1:10)	0.2172	0.2913	0.2225	0.2220	0.2355
Prosper (1:20)	0.2327	0.3144	0.2268	0.2257	0.2255
Prosper (1:40)	0.2230	0.3517	0.2263	0.2366	0.2237
Prosper (1:60)	0.2401	0.3248	0.2274	0.2444	0.2334
Prosper (1:80)	0.2270	0.2651	0.2264	0.2617	0.2309
Prosper (1:100)	0.2268	0.2936	0.2275	0.2667	0.2377
Home	0.2149	0.2138	0.2092	0.2145	0.2058
Home (1:4)	0.2280	0.2207	0.2140	0.2179	0.2161
Home (1:6)	0.2375	0.2204	0.2160	0.2182	0.2173
Home (1:8)	0.2283	0.2223	0.2168	0.2187	0.2123
Home (1:10)	0.2069	0.2426	0.2181	0.2191	0.2095
Home (1:20)	0.2152	0.2350	0.1925	0.2018	0.2175
Home (1:40)	0.2173	0.2465	0.2075	0.2279	0.2112
Home (1:60)	0.2207	0.3003	0.2131	0.2381	0.2126
Home (1:80)	0.2245	0.2481	0.2138	0.2535	0.2171
Home (1:100)	0.2232	0.2501	0.2125	0.2625	0.2130

References

Altmann, A., Toloşi, L., Sander, O., & Lengauer, T. (2010). Permutation importance: A corrected feature importance measure. *Bioinformatics*, *26*(10), 1340–1347.

Blagus, R., & Lusa, L. (2012). Evaluation of SMOTE for high-dimensional class-imbalanced microarray data. In *2012 11th international conference on machine learning and applications* (pp. 89–94). IEEE.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *The Journal of Artificial Intelligence Research*, *16*, 321–357.

Crone, S. F., & Finlay, S. (2012). Instance sampling in credit scoring: An empirical study of sample size and balancing. *International Journal of Forecasting*, *28*(1), 224–238.

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*(1), 1–30.

Douzas, G., & Bacao, F. (2018). Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with Applications*, *91*, 464–471.

Durand, D. (1941). *Risk elements in consumer instatement financing*. Cambridge, MA: National Bureau of Economic Research.

Engelmann, J., & Lessmann, S. (2021). Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning. *Expert Systems with Applications*, *174*, 1–13.

Fajardo, V. A., Findlay, D., Houmanfar, R., Jaiswal, C., Liang, J., & Xie, H. (2018). VOS: A method for variational oversampling of imbalanced data. arXiv:1809.02596.

Fiore, U., De Santis, A., Perla, F., Zanetti, P., & Palmieri, F. (2019). Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, *479*, 448–455.

Freedman, S., & Jin, G. Z. (2017). The information value of online social networks: Lessons from peer-to-peer lending. *International Journal of Industrial Organization*, *51*, 185–222.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, *32*, 675–701.

García, S., & Herrera, F. (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, *9*(12), 2677–2694.

García, V., Marqués, A., & Sánchez, J. S. (2012). On the use of data filtering techniques for credit risk prediction with instance-based models. *Expert Systems with Applications*, *39*(18), 13267–13276.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 315–323). JMLR Workshop and Conference Proceedings.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, *46*(1), 389–422.

Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing* (pp. 878–887). Springer.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 international joint conference on neural networks* (pp. 1322–1328). IEEE.

Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics. Theory and Methods*, *9*(6), 571–595.

Khan, F. N., Khan, A. H., & Israt, L. (2020). Credit card fraud prediction and classification using deep neural network and ensemble learning. In *2020 IEEE region 10 symposium* (pp. 1–6). IEEE.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv:1312.6114.

Kulkarni, T. D., Whitney, W. F., Kohli, P., & Tenenbaum, J. (2015). Deep convolutional inverse graphics network. In *29th annual conference on neural information processing systems* (pp. 2539–2547). NIPS.

Kullback, S. (1997). *Information theory and statistics*. Mineola, NY: Dover Publications.

Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, *247*, 124–136.

Levi, G., & Hassner, T. (2015). Age and gender classification using convolutional neural networks. In *2015 IEEE conference on computer vision and pattern recognition workshops* (pp. 34–42). IEEE Computer Society.

Li, W., Ding, S., Chen, Y., & Yang, S. (2018). Heterogeneous ensemble for default prediction of peer-to-peer lending in China. *IEEE Access*, *6*, 54396–54406.

Li, S. C., Tai, B. C., & Huang, Y. (2019). Evaluating variational autoencoder as a private data release mechanism for tabular data. In *2019 IEEE 24th pacific rim international symposium on dependable computing* (pp. 198–1988). IEEE.

Liu, Z., & Pan, S. (2018). Fuzzy-rough instance selection combined with effective classifiers in credit scoring. *Neural Processing Letters*, *47*(1), 193–202.

Ma, C., Liu, Z., Cao, Z., Song, W., Zhang, J., & Zeng, W. (2020). Cost-sensitive deep forest for price prediction. *Pattern Recognition*, *107*, 1–16.

Mangasarian, O. L. (1965). Linear and nonlinear separation of patterns by linear programming. *Operations Research*, *13*(3), 444–452.

Marqués, A. I., García, V., & Sánchez, J. S. (2013). On the suitability of resampling techniques for the class imbalance problem in credit scoring. *Journal of the Operational Research Society*, *64*(7), 1060–1070.

Mpofu, T. P., & Mukosera, M. (2014). Credit scoring techniques: A survey. *International Journal of Science and Research*, *3*(8), 165–168.

Nanni, L., Fantozzi, C., & Lazzarini, N. (2015). Coupling different methods for overcoming the class imbalance problem. *Neurocomputing*, *158*, 48–61.

Odaibo, S. (2019). Tutorial: Deriving the standard variational autoencoder (VAE) loss function. arXiv:1907.08956.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning* (pp. 1278–1286). PMLR.

Salimans, T., Kingma, D., & Welling, M. (2015). Markov chain Monte Carlo and variational inference: Bridging the gap. In *32nd international conference on machine learning* (pp. 1218–1226). IMLS.

Shen, F., Zhao, X., Kou, G., & Alsaadi, F. E. (2021). A new deep learning ensemble credit risk evaluation model with an improved synthetic minority oversampling technique. *Applied Soft Computing*, *98*, 1–14.

Sigmon, K., & Davis, T. A. (2004). *MATLAB primer*. New York: Chapman and Hall/CRC.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(1), 1929–1958.

Su, R., Liu, X., & Wei, L. (2020). MinE-RFE: Determine the optimal subset from RFE by minimizing the subset-accuracy–defined energy. *Briefings in Bioinformatics*, *21*(2), 687–698.

Sun, J., Lang, J., Fujita, H., & Li, H. (2018). Imbalanced enterprise credit evaluation with DTE-SBD: Decision tree ensemble based on SMOTE and bagging with differentiated sampling rates. *Informing Science*, *425*, 76–91.

Tan, F., Hou, X., Zhang, J., Wei, Z., & Yan, Z. (2019). A deep learning approach to competing risks representation in peer-to-peer lending. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(5), 1565–1574.

Thomas, L. C. (2000). A survey of credit and behavioural scoring: Forecasting financial risk of lending to consumers. *International Journal of Forecasting*, *16*(2), 149–172.

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.

Wallace, B. C., & Dahabreh, I. J. (2014). Improving class probability estimates for imbalanced data. *Knowledge and Information Systems*, *41*(1), 33–52.

Wan, Z., Zhang, Y., & He, H. (2017). Variational autoencoder based synthetic data generation for imbalanced learning. In *2017 IEEE symposium series on computational intelligence* (pp. 1–7). IEEE.

Wang, C., Han, D., Liu, Q., & Luo, S. (2018). A deep learning approach for credit scoring of peer-to-peer lending using attention mechanism LSTM. *IEEE Access*, *7*, 2161–2168.

Wang, Y., Jia, Y., Tian, Y., & Xiao, J. (2022). Deep reinforcement learning with the confusion-matrix-based dynamic reward function for customer credit scoring. *Expert Systems with Applications*, *200*, Article 117013.

Xiao, J., Cao, H., Jiang, X., Gu, X., & Xie, L. (2017). GMDH-based semi-supervised feature selection for customer classification. *Knowledge-Based Systems*, *132*(15), 236–248.

Xiao, J., Jia, Y., Jiang, X., & Wang, S. (2020). Circular complex-valued GMDH-type neural network for real-valued classification problems. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(12), 5285–5299.

Xiao, J., Tian, Y., Xie, L., Jiang, X., & Huang, J. (2019). A hybrid classification framework based on clustering. *IEEE Transactions on Industrial Informatics*, *16*, 2177–2188.

Xiao, J., Wang, Y., Chen, J., Xie, L., & Huang, J. (2021). Impact of resampling methods and classification models on the imbalanced credit scoring problems. *Information Sciences*, *569*, 508–526.

Xiao, J., Zhou, X., Zhong, Y., Xie, L., Gu, X., & Liu, D. (2020). Cost-sensitive semi-supervised selective ensemble model for customer credit scoring. *Knowledge-Based Systems*, *189*, 105–118.

Yu, L., Yang, Z., & Tang, L. (2016). A novel multistage deep belief network based extreme learning machine ensemble learning paradigm for credit risk assessment. *Flexible Services and Manufacturing Journal*, *28*(4), 576–592.

Zhang, C., Zhou, Y., Chen, Y., Deng, Y., Wang, X., Dong, L., & Wei, H. (2018). Over-sampling algorithm based on VAE in imbalanced classification. In *International conference on cloud computing* (pp. 334–344). Springer.

Zhou, Z. H., & Feng, J. (2019). Deep forest. *National Science Review*, *6*(1), 74–86.

Zhu, B., Yang, W. C., Wang, H. X., & Yuan, Y. (2018). A hybrid deep learning model for consumer credit scoring. In *2018 international conference on artificial intelligence and big data* (pp. 205–208). IEEE.