Artificial Intelligence Review (2024) 57:93
https://doi.org/10.1007/s10462-023-10697-9

Deep reinforcement learning based on balanced stratified prioritized experience replay for customer credit scoring in peer-to-peer lending

Yadong Wang[1] · Yanlin Jia[2] · Sha Fan[3] · Jin Xiao[3]

Published online: 18 March 2024

Abstract
In recent years, deep reinforcement learning (DRL) models have been successfully utilised to solve various classification problems. However, these models have never been applied to customer credit scoring in peer-to-peer (P2P) lending. Moreover, the imbalanced class distribution in experience replay, which may affect the performance of DRL models, has rarely been considered. Therefore, this article proposes a novel DRL model, namely a deep Q-network based on a balanced stratified prioritized experience replay (DQN-BSPER) model, for customer credit scoring in P2P lending. Firstly, customer credit scoring is formulated as a discrete-time finite-Markov decision process. Subsequently, a balanced stratified prioritized experience replay technology is presented to optimize the loss function of the deep Q-network model. This technology can not only balance the numbers of minority and majority experience samples in the mini-batch by using stratified sampling technology but also select more important experience samples for replay based on the priority principle. To verify the model performance, four evaluation measures are introduced for the empirical analysis of two real-world customer credit scoring datasets in P2P lending. The experimental results show that the DQN-BSPER model can outperform four benchmark DRL models and seven traditional benchmark classification models. In addition, the DQN-BSPER model with a discount factor γ of 0.1 has excellent credit scoring performance.

Keywords Customer credit scoring · Deep reinforcement learning · Deep Q-network · Experience replay · P2P lending

✉ Jin Xiao
xiaojin@scu.edu.cn

Yadong Wang
wangyadong123.wang@163.com

Yanlin Jia
yelinjyl@126.com

Sha Fan
fanshahn@163.com

[1] Public Administration School, Guangzhou University, Guangzhou 510006, Guangdong, China

[2] School of Science, Southwest Petroleum University, Chengdu 610500, Sichuan, China

[3] Business School, Sichuan University, Chengdu 610064, Sichuan, China

1 Introduction

In recent years, P2P lending companies are rapidly developing, mainly including Leading Club, Kiva and Zopa, etc. Further, P2P lending has gradually become an important channel for small loans and private financing. However, due to network virtualisation and imperfection monitoring, P2P lending entails greater credit risks than traditional bank lending (Du et al. 2020). Customer credit scoring (CCS) is an effective tool for assessing credit risk in P2P lending. Generally, CCS can be regarded as a binary classification problem in which customer credit is divided into two categories: 'good' and 'bad'.

At present, there are three types of credit scoring methods: expert judgement, statistical analysis, and machine learning (Baesens et al. 2003; Dastile et al. 2020; Xia et al. 2020). Common expert judgement methods include 5C and 5P, which rely on expert experience to evaluate customer credit. Statistical analysis methods have been proposed to improve the efficiency of CCS, including linear discriminant analysis (LDA; Altman 1968) and logistic regression (LR; Hosmer et al. 2013) methods. Machine learning methods mainly include naive Bayes (NB; Rish 2001), decision tree (DT; Yeo and Grant 2018), k-nearest neighbour (KNN; Wauters and Vanhoucke 2017), support vector machine (SVM; Trafalis and Gilbert 2006), and deep neural network (DNN; Gunnarsson et al. 2021). The experimental results of some studies have shown that machine learning methods often achieve better credit scoring performance than statistical analysis methods (Blumenstock et al. 2022; Dumitrescu et al. 2022; Lessmann et al. 2015; Petrides et al. 2020; Serrano-Cinca and Gutiérrez-Nieto 2016).

With the deepening of research, it has been found that the class distribution of credit scoring datasets in P2P lending is often highly imbalanced; that is, the samples of customers with bad credit are much smaller than those with good credit, which may lead to poor classification accuracy in credit scoring for customers with bad credit (Crone and Finlay 2012; Veganzones and Séverin 2018; Xiao et al. 2021). To solve this problem, resampling methods [random oversampling (ROS), random undersampling (RUS), etc.] are proposed to balance the class distribution of the training set before modelling (Marqués et al. 2013; Protopapadakis et al. 2019).

When using the traditional classification model for CCS in P2P lending, the assumptions utilised in the existing studies are that the samples in the dataset are independent and identically distributed (Borgonovo and Smith 2011; Lopez-Martin et al. 2020; Óskarsdóttir et al. 2019). However, for real-world credit scoring in P2P lending, a large number of samples are generated through the dynamic interaction between customers and financial institutions or platforms, which are not strictly independent and identically distributed. The developed deep reinforcement learning (DRL) model provides a new method of solving the above problems, it is a dynamic decision-making method based on the Markov decision process (MDP; Mnih et al. 2015). Thus far, DRL models have been successfully used in various fields, including management strategy optimisation (Liu et al. 2020; Schnaubelt 2022; van Heeswijk 2022), energy management (Sun 2020), and autopilot technology (Wurman et al. 2022).

Among the various DRL models, the deep Q-network (DQN) model (Mnih et al. 2013) is the most commonly used. Therefore, some scholars have attempted to employ DQN models to solve classification problems (Chatterjee and Namin 2019; Ding et al. 2019; Li and Xu 2020; Martinez et al. 2020; Wang et al. 2022; Zhao et al. 2016). The core idea is firstly to formulate the classification problem as an MDP and then to use experience replay technology to build the mini-batch in the training set to optimize the loss function of the

DQN model dynamically. Finally, the optimized DQN model is applied to classify the samples in the test set. In particular, DQN models have gradually shown their advantages in binary classification (Lin et al. 2020; Lopez-Martin et al. 2020; Martinez et al. 2020). The most commonly used experience replay technology in DQN models is random experience replay (RER), which uses a random sampling method to select experience samples from the buffer to build the mini-batch. However, RER technology has difficulty converging DQN models in complex scenarios. Therefore, stratified experience replay (SER) technology (Chen et al. 2018) and prioritized experience replay (PER) technology (Schaul et al. 2015) have been developed to improve the convergence performance of DQN models.

The previous studies have significantly contributed to the application of DRL models in classification tasks. However, the methods employed have limitations. First, mini-batches have mainly been constructed based on RER, SER, or PER to optimize the loss functions of DQN models in the previous research. Although these experience replay technologies can improve the convergence performance of DQN models to some extent, balancing the numbers of minority and majority experience samples in the mini-batch is difficult, which may affect the classification performance of DQN models, especially in imbalanced classification. Second, when applying DQN models for classification, most scholars have designed the value of the discount factor according to common DRL environments (such as autopilot, robot control, and computer games), considering their effects on the credit scoring performance of DQN models. If an inappropriate discount factor value is designed, the DQN model performance may worsen. Third, the DQN-IRF model proposed by Lin et al. (2020) addresses classification tasks with imbalanced class distributions. However, this model is mainly applied to image and text classification. The features of these samples for classification are very different from those of samples for CCS, which can lead to the poor credit scoring performance of the DQN-IRF model.

To solve the above problems, we constructed a DQN based on the balanced stratified prioritized experience replay (DQN-BSPER) model. Firstly, we formulated CCS as a discrete-time finite MDP according to the characteristics of credit scoring. Then, we developed balanced stratified prioritized experience replay (BSPER) technology to improve the experience replay process of DQN models to optimize the model loss function. To verify the model performance, we introduced four evaluation measures (EMs) for empirical analysis on two real-world CCS datasets in P2P lending with an imbalanced class distribution. Firstly, the effects of the discount factor and proportion of minority and majority experience samples in the stratified priority mini-batch on the CCS performance of the DQN-BSPER model were analysed. Then, we compared and analysed the CCS performance of the DQN-BSPER model and four other benchmark DRL models, namely, a DQN, DQN based on stratified experience replay (DQN-SER), DQN based on prioritized experience replay (DQN-PER), and DQN-IRF, and further compared their convergence performance. Next, the CCS performance of the DQN-BSPER model was statistically compared with those of seven traditional benchmark classification models. Finally, the effects of the imbalanced class distribution on the CCS performance of the DQN-BSPER model were analysed.

The main contributions of this paper are as follows:

(1) we propose the BSPER technology to improve the experience replay process of the DQN model for CCS in P2P lending. The proposed BSPER technology can not only reduce the impact of the highly imbalanced class distribution on the DQN model performance by balancing the numbers of minority and majority samples in the mini-

batch, but also select more important experience samples according to the temporal difference (TD) error to improve the convergence performance of the DQN model.
(2) The effects of the discount factor on the CCS performance of the DQN-BSPER model are analysed, compensating for the fact that previous scholars have often designed the discount factor according to common DRL environments (such as autopilot, robot control, and computer games), which may lead to poor CCS performance.
(3) It verifies that the proposed DQN-BSPER model exhibits excellent CCS performance in P2P lending, demonstrating that it could serve as a credit scoring tool in P2P lending for financial institutions.

The remainder of this paper is organised as follows. Section 2 provides a literature review. The details of the theoretical background are described in Sect. 3. Section 4 elaborates the discrete-time finite MDP for CCS. Section 5 introduces the process of the DQN-BSPER model in detail. The experimental design is elaborated in Sect. 6. Section 7 presents the experimental result. Finally, conclusions and future works are given in Sect. 8.

2 Literature review

2.1 Customer credit scoring

In actual CCS, banks or financial institutions determine whether to grant loans to customers based on customer credit. Even if customers have different credit grades, the final result is still 'granting' or 'not granting'. Therefore, CCS can be regarded as a binary classification problem.

Currently CCS models stemming from operations research and artificial intelligence have also become popular, include LR (Hosmer et al. 2013), NB (Rish 2001), DT (Yeo and Grant 2018), KNN (Wauters and Vanhoucke 2017), SVM (Trafalis and Gilbert 2006), and DNN (Gunnarsson et al. 2021) models. For instance, Lessmann et al. (2015) compared the 41 classification models performance on eight CCS data sets and validated that the DNN model achieved excellent performance. Fernandes and Artes (2016) introduced a measure of the local default risk based on the application of ordinary kriging to logistic credit scoring models. These models achieved better performance on the Brazilian dataset. Li et al. (2020) proposed a recursive Bayes estimator to improve the precision of credit scoring by incorporating the dynamic interaction topology of customers. The experimental results showed that, under the proposed framework, the designed estimator achieved a higher precision than any efficient estimator. Xiao et al. (2021) compared DNN, LDA, LR, DT, and SVM models performance. The experimental results on seven CCS datasets showed that the LR model provided the best performance. Wang et al. (2022) proposed an innovative DQN model for CCS and compared the performance of the proposed DQN model with those of eight other classification models. The experimental results obtained using five CCS datasets showed that the proposed model performed significantly better than the other eight traditional classification models.

In recent years, P2P lending has developed rapidly with the rise of the Internet, and scholars have begun to focus on customer credit scores in P2P lending. For instance, Guo et al. (2016) designed an instance-based credit scoring model that can assess the return and risk of loans. To verify the proposed model, the authors conducted extensive experiments on two actual CCS datasets in P2P lending. The experimental results showed that

this model could effectively improve investment performance. Wang et al. (2021) proposed a misclassification cost matrix for P2P credit grading, using a set of equations and models to calculate costs. The results obtained on the Lending Club dataset showed that cost-sensitive classifiers could significantly reduce the total cost. Bastani et al. (2019) proposed a two-stage scoring method based on credit and profit. The first stage identifies the NPLs. The second stage predicts profitability based on the internal rate of return. In both stages, wide and deep learning were used to build the prediction models. The results obtained using the Lending Club dataset showed that the proposed model outperformed the existing credit scoring and profit scoring methods.

In summary, in real-world credit scoring, especially in P2P lending, data are generated in the dynamic interaction between customers and financial institutions, which means that a sequence correlation may exist between the samples, which can affect the CCS performance of the traditional classification model. The DQN-BSPER model proposed in this paper is a sequential decision model and verifies whether there is a sequence correlation between samples in the CCS datasets in P2P lending.

2.2 Deep reinforcement learning for classification problems

DRL has been widely used in various real-world fields (Moor et al. 2022; Fan et al. 2020; Liu et al. 2020; Patel et al. 2019; Silver et al. 2018), and the most popular model is the DQN model (Mnih et al. 2013, 2015). Over the years, increasingly many scholars have begun using the DQN model for supervised classification. For instance, Zhao et al. (2016) proposed a DQN model for image classification and experimentally proved that the proposed model was highly competitive on the vehicle classification datasets. Lin et al. (2020) proposed a DQN based on an improved reward function (DQN-IRF) model. The model provides more rewards to minority experience samples to make the classification strategy more inclined toward the minority, which effectively improves the classification performance of the model when applied to image and text datasets.

In binary classification, the DQN model have shown excellent performance. For instance, Lopez-Martin et al. (2020) used four DRL models for intrusion detection and verified that their performance were better than traditional classification models. Ding et al. (2019) constructed a DQN model with RER technology to identify machinery running faults and achieved 100% recognition accuracy. Lim et al. (2021) used a DQN model with RER technology to intelligently predict the hidden relationships in criminal network. The experimental results showed that the proposed model performance was superior to RF and SVM. Lin et al. (2020) proposed a DQN model with RER technology for classification task through making the results inclined toward the minority class, and then verify that the improved DQN model was significantly superior to the DNN model on the image and text datasets. In addition, Chen et al. (2018) introduced stratified sampling technology into the experience replay process of the DQN model to improve its convergence performance. Schaul et al. (2015) introduced the degree of importance (referred to as priority) into the experience replay process and developed PER technology. This technology firstly determines the priority of each experience sample according to the TD error. Subsequently, it selects important experience samples to construct the mini-batch based on the priority to optimize the loss function. They proved that the improved process is very effective in improving convergence for DRL models.

In summary, previous scholars have mainly constructed mini-batches based on RER, SER, or PER for DQN models. It is difficult to balance the numbers of minority and majority experience samples in the mini-batch, which may affect the DQN model performance in imbalanced classification. Our proposed BSPER technique fully considers the class-imbalanced characteristics of the CCS datasets in P2P lending and improves the convergence performance of DQN.

3 Theoretical background

3.1 Notations

For convenience and clarity, the main mathematical notations and definitions used in this article are presented in "Appendix 1".

3.2 Reinforcement learning and Q-learning

Reinforcement learning (RL) is a subclass of machine learning that aims to optimize the action strategy of the agent continuously to maximise the expected cumulative reward in the process of interaction with the environment, that is, to maximise the Q-function (Sutton and Barto 1998). In previous studies, RL tasks have usually been formulated as MDPs (Cai et al. 2020; Zhang et al. 2021), and their basic elements can be expressed as a tuple $(\boldsymbol{S}, \boldsymbol{A}, P, R, \gamma)$, where $\boldsymbol{S}$ indicates the state space, $\boldsymbol{A}$ indicates the action space, $P : \boldsymbol{S} \times \boldsymbol{A} \times \boldsymbol{S} \to [0, 1]$ indicates the state transition probability, $R : \boldsymbol{S} \times \boldsymbol{A} \to \mathbb{R}$ indicates the reward function, and $\gamma \in [0, 1]$ indicates the discount factor that balances the importance of future rewards and current rewards. In particular, during the tth ($t \in [0, T]$) time step, the agent first performs an action $a_t \in \boldsymbol{A}$ under the environment state $s_t \in \boldsymbol{S}$. Then, the reward r_t is generated by environment and feeds it back to the agent. Finally, the environment is transferred to the next state s_{t+1} according to probability P. The cumulative reward from s_t to s_T can be expressed as $R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$. In addition, the expected cumulative reward (usually represented by the Q-function) corresponding to the state–action pair $\left(s_t, a_t\right)$ is expressed as $Q\left(s_t, a_t\right) = E[\sum_{i=t}^{T} \gamma^{i-t} r_i]$ (Chen et al. 2018), and the optimal strategy π^* represents the strategy that can maximise the Q-function. According to the Bellman equation (Gosavi 2009), the Q-function can be transformed into the following form:

$$Q\left(s_t, a_t\right) = E\left[r_t + \gamma \max_{a_{t+1} \in \boldsymbol{A}} Q\left(s_{t+1}, a_{t+1}\right)\right], \tag{1}$$

where $\max_{a_{t+1} \in \boldsymbol{A}} Q\left(s_{t+1}, a_{t+1}\right)$ represents the maximum Q-value that the agent can obtain when the state is s_{t+1}.

Q-learning is a widely used model-free RL algorithm based on asynchronous dynamic programming that can quickly find the optimal strategy for the MDP (Watkins and Dayan 1992). The core idea is to find the optimal strategy π^* that can maximise the Q-value using the Bellman equation (Gosavi 2009) to iterate the Q-table continuously. The general steps of Q-learning can be summarised as follows:

Step 1: Initialise the Q-values of all state-action pairs in the Q-table.

Step 2: According to the Q-table, the agent executes action a_t in state s_t.

Step 3: When the state-action pair is (s_t, a_t), the agent obtains reward r_t according to the reward function. Simultaneously, the next state s_{t+1} is generated from environment, then iteratively updates the Q-value using the Bellman equation (Gosavi 2009):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a_{t+1} \in \boldsymbol{A}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right], \tag{2}$$

where α represents the learning rate.

Step 4: Repeat Steps 2 and 3 until none of the Q-values in the Q-table change. Then, the strategy corresponding to the Q-table is the optimal strategy π^* (Sutton and Barto 1998).

3.3 Deep Q-network

Q-learning algorithms have been successfully applied in many fields of real word, but they are primarily suitable for tasks with small state spaces. In the real world, the state spaces of tasks are typically large, and the number of states can even reach tens of millions. To solve this problem, Mnih et al. (2015) combined a DNN with Q-learning to develop a DQN model, which approximately expresses the Q-function by automatically extracting features from the state space. Specifically, experience samples are continuously obtained first according to the greedy strategy, and they are stored in a fixed-size replay memory buffer to form an experience sample set. In particular, the experience sample at the tth time step is represented as $e_t = (s_t, a_t, r_t, s_{t+1})$, and the experience sample set is represented as $\boldsymbol{E}_t = \{e_1, e_2, \ldots, e_t\}$. Then, k experience samples are randomly selected from $\boldsymbol{E}_t$ to form a mini-batch using RER technology. Finally, a DNN is used to fit the Q-function, so the network corresponding to the Q-function is called the Q-network, and its general expression is $Q(s, a; \theta)$, whereas the loss function of $L(\theta_t^C)$ can be expressed as

$$L(\theta_t^C) = \frac{1}{k} \sum_{i=1}^{k} \left(y_i - Q(s_i, a_i; \theta_t^C) \right)^2, \tag{3}$$

where $y_i = r_i + \gamma \max_{a_{i+1} \in \boldsymbol{A}} Q(s_{i+1}, a_{i+1}; \theta_t^T)$ indicates the target Q-function; θ_t^T indicates the parameters of the target Q-network; $Q(s_i, a_i; \theta_t^C)$ indicates the current Q-function; θ_t^C indicates the parameters of the current Q-network. The gradient descent algorithm was used to update the parameters of the Q-network:

$$\frac{\nabla L(\theta_t^C)}{\nabla \theta_t^C} = \frac{2}{k} \sum_{i=1}^{k} \left(y_i - Q(s_i, a_i; \theta_t^C) \right) \frac{\nabla L\left(Q(s_i, a_i; \theta_t^C) \right)}{\nabla \theta_t^C}, \tag{4}$$

$$\theta_{t+1}^C = \theta_t^C - \alpha \frac{\nabla L(\theta_t^C)}{\nabla \theta_t^C}, \tag{5}$$

where α indicates the learning rate of the Q-network. After training, we can obtain the optimal strategy $\pi^* = \underset{a \in A}{argmax} Q^*(s, a; \theta)$, which can maximise the Q-function. In addition, iterative updating technology is used to reduce the correlation between the current Q-network and target Q-network. In other words, the parameter of the current Q-network θ_t^C is assigned to the parameter of the target Q-network θ_t^T after a certain number of steps for improving the convergence stability of the Q-network (Lopez-Martin et al. 2020; Mnih et al. 2015). For the detailed algorithm, see the literature (Mnih et al. 2015).

3.4 Experience replay

In most DRL frameworks, the agent constantly receives new experience samples to update the parameters of the DQN model incrementally. The simplest update method uses only one experience sample in each time step to update the model parameters. However, the biggest drawback of this method is that the rare experiences that may be useful in the future will be quickly forgotten, resulting in low sampling efficiency. Experience replay can effectively solve this issue (Luo et al. 2018). In other words, experience samples are firstly stored in a fixed-size replay memory buffer; then, each time the parameters of the DQN model are updated, a fixed number of experience samples are sampled from the replay memory buffer to construct a mini-batch; and finally, the gradient descent algorithm is used to train and optimize the model.

Obviously, a complete experience replay process consists of storing and sampling the experience samples. Therefore, the selection of experience samples from the replay memory buffer is important in improving the DQN model performance. In particular, when an experience sample is stored in replay memory buffer D, a new index label $i \in \{1, 2, \ldots, d\}$ is assigned to the experience. The priority of the ith experience sample is represented as $P(e_i)$. The entire replay memory buffer can be regarded as a combination of experience samples and priority $\{e_i, P(e_i)\}$. The key to selecting the experience samples is to determine $P(e_i)$ for each experience sample. The most commonly used experience replay technology in the DQN model is RER, and the priority of each experience sample is the same, that is, $P(e_i) = \frac{1}{d}$. This technology uses a random sampling method to select experience samples from the buffer to construct the mini-batch, which ignores the experience samples that play an important role in the parameter updating of the DQN model. Consequently, the DQN model may converge slowly in complex tasks. To solve this problem, Schaul et al. (2015) proposed PER technology. The core steps of this technology can be summarised as follows. Firstly, the TD error of the experience sample in the replay memory buffer can be calculated as follows:

$$\delta_i = \left| y_i - Q(s_i, a_i; \theta_t^C) \right|, i = 1, 2, ..., d. \tag{6}$$

The priority of each experience sample is then determined according to the TD error; that is, the selected probability of the experience sample can be expressed as

$$P(e_i) = \frac{\delta_i}{\sum_1^d \delta_i}. \tag{7}$$

Finally, k experience samples are selected from the replay memory buffer according to the probability $P(e_i)$ to construct a mini-batch. The higher the TD error, the higher the priority of the experience sample. Thus, a sample with a higher TD error is selected with

a higher probability, which effectively improves the convergence performance of the DQN model (Schaul et al. 2015).

Furthermore, Chen et al. (2018) developed an SER by introducing stratified sampling technology into the experience replay for the DQN model. The core idea of this technology is to use a stratified sampling method to select different classes of experience samples from the replay memory buffer to construct a mini-batch. The experimental results demonstrate that this technology can dramatically improve the classification performance of the DQN model.

4 Formulating customer credit scoring in P2P lending as discrete-time finite MDP

In the real world, CCS in P2P lending is a interactive process similar to the RL process. The number of environmental states is limited within a certain time step, and the time step is discrete. Therefore, we formulated CCS in P2P lending as a discrete-time finite MDP (CSDFMDP). Firstly, a limited number of customers enter the environment. The agent then identifies the environmental state and performs a credit scoring action according to the state. Next, the environment generates rewards based on the reward function and feeds them back to the agent. Finally, the agent optimizes the action strategy based on the feedback reward. This process is repeated until there are no more customers in the environment. The ultimate objective of the agent is to classify the customer samples as accurately as possible. More importantly, the agent can obtain different rewards when classifying the customer samples correctly or incorrectly. Therefore, the agent can optimize the credit scoring action by maximising the Q-function.

To describe the CSDFMDP more clearly, we use $\boldsymbol{D}_{train} = \{(x_t, y_t)\}_{1 \le t \le T}$ to represent the CCS training set, where T indicates the number of customer samples, $x_t = \left(x_t^1, x_t^2, \ldots, x_t^g\right)$ indicates the tth customer sample, g indicates the number of features, and $y_t \in \{0, 1\}$ indicates the class label of x_t. The basic elements related to CSDFMDP can be described as follows:

(1) **Environment**. In the real world, one of the most critical factors affecting the P2P CCS environment is the customer, which was the main object of our study. Then, we simplified the CSDFMDP environment to include only the customer.
(2) **Environment state space**. The environment state space can be expressed as $\boldsymbol{S} = \left\{s_1, s_2, \ldots, s_T\right\}$, where the environment state at the tth time step is defined as $s_t = \left(s_t^1, s_t^2, \ldots, s_t^g\right)$. In particular, s_1 indicates the initial environment state corresponding to x_1 in the training set, and s_T indicates the terminal environment state corresponding to x_T in the training set.
(3) **Agent**. The agent represents a substitute for the bank loan approver, which classifies customer credit according to the environmental state.
(4) **Action space**. The action space is represented as $\boldsymbol{A} = \{0, 1\}$, where 0 and 1 indicates that the agent classifies the customer as having good credit and bad credit respectively. Then, at the tth time step, the credit scoring action performed by the agent according to state s_t is expressed as $a_t \in \boldsymbol{A}$.
(5) **Reward**. At the tth time step, the feedback reward from the environment is r_t, that is, if the agent classifies the customer credit correctly, then the environment feeds back a positive reward to the agent according to the reward function; otherwise, the environ-

ment feeds back a negative reward. Referring to the literature (Chatterjee and Namin 2019; Lopez-Martin et al. 2020), we set the reward function for CCS as follows:

$$R(a_t, y_t) = \begin{cases} 1, a_t = y_t, \\ -1, a_t \neq y_t, \end{cases} \tag{8}$$

where $R(a_t, y_t)$ indicates that if the credit scoring action of the agent a_t is the same as the class label of customer credit y_t, then the feedback reward from the environment is $r_t = 1$; otherwise, it is $r_t = -1$.

(6) **State transition probabilities**. According to the literature (So and Thomas 2011), the state transition probability of the CSDFMDP is as follows:

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_1, a_1), \tag{9}$$

where $p(s_{t+1}|s_t, a_t)$ indicates the probability of the environment transferring to state s_{t+1} when the state–action pair is (s_t, a_t). In particular, the order of the customer samples was fixed; therefore, the state transition probability was deterministic, that is, $p(s_{t+1}|s_t, a_t) = 1$.

(7) **Strategy**. In the CSDFMDP, the classification strategy $\pi(a_t|s_t)$ indicates the probability of performing credit scoring action a_t by the agent under environment state s_t, so the optimal credit scoring action according to the greedy strategy can be expressed as follows:

$$\pi^*(a_t|s_t) = \begin{cases} 1, \, if \, a_t = \underset{a_t \in A}{argmax} Q(s_t, a_t), \\ 0, else, \end{cases} \tag{10}$$

where the greedy strategy means that the agent only selects the action that maximises $Q(s_t, a_t)$ in environment state s_t.

Specifically, the process of the agent from s_t to s_T is as follows.

Step 1: When the CCS environment state is s_t, the agent performs a credit scoring action based on greedy policy a_t.

Step 2: According to the reward function $R(a_t, y_t)$, the environment feeds back a reward to the agent; that is, if the credit scoring action of agent a_t is the same as real class label y_t, then the feedback reward from the environment is $r_t = 1$; otherwise, it is $r_t = -1$.

Step 3: According to the state–action pair (s_t, a_t), the environment is transferred to the next state, s_{t+1}.

Step 4: Repeat Steps 1–3 until the environment reaches the terminal state, s_T.

The cumulative reward obtained by the agent from s_t to s_T is $R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$. The Q-function corresponding to the state-action pair can be expressed as $(s_t, a_t) = E\left[r_t + \gamma \max_{a_{t+1} \in A} Q(s_{t+1}, a_{t+1})\right]$, where $\gamma \in [0, 1]$ indicates the discount factor. In particular, $\gamma = 0$ indicates that only the current credit scoring result is considered, $\gamma = 1$ indicates that the future and current credit scoring results both have the equal importance.

5 Proposed DQN based on balanced stratified prioritized experience replay for customer credit scoring in P2P lending

This section proposes the DQN-BSPER model for CCS in P2P lending. We firstly combined SER and PER to develop the BSPER technology and then designed the DQN-BSPER model; that is, we used the BSPER technology to improve the experience replay process of the DQN model to optimize its loss function. This section introduces the BSPER technology in detail and describes the process of using the DQN-BSPER model for CCS.

5.1 Balanced stratified prioritized experience replay

The most commonly used experience replay technology in the DQN model is RER, which randomly selects experience samples from the buffer to construct a mini-batch. The main advantage of this method is that it is easy to implement. However, the mini-batch constructed by RER technology has an imbalanced class distribution in CCS, and it is difficult to select experience samples that play important roles in updating the loss function parameters, which may affect the CCS performance of the DQN model. SER can reduce the effects of the imbalanced class distribution on the DQN model performance by adjusting the numbers of minority and majority experience samples in the mini-batch (Chen et al. 2018). However, it is difficult to select experience samples that play important roles in updating the loss function parameters. To avoid the shortcomings of the SER, PER provides a concept that can select more important experience samples to improve the convergence performance of the DQN model (Schaul et al. 2015), but the class distribution of the constructed mini-batch is still imbalanced. It can be observed that SER and PER are complementary in the experience replay process. Therefore, we combined SER and PER to develop the BSPER technology.

To express the sampling process of the BSPER technology more clearly, we provide definitions of the majority and minority experience samples. If the current state of an experience sample is a minority sample (a positive class sample), then the sample is the minority experience sample, which is represented as $\boldsymbol{E}^{min}$, and the tth minority experience sample is represented as $e_t^{min} = (s_t^{min}, a_t^{min}, r_t^{min}, s_{t+1}^{min})$. If the current state of an experience sample is a majority class sample (a negative class sample), then let the sample be the majority experience sample, which is represented as $\boldsymbol{E}^{maj}$, and let the tth majority experience sample be indicated as $e_t^{maj} = (s_t^{maj}, a_t^{maj}, r_t^{maj}, s_{t+1}^{maj})$, $\boldsymbol{E}^{min} \cap \boldsymbol{E}^{maj} = \varnothing$. The core steps of the SPER technology can be described as follows.

Step 1: Store the experience samples in the minority and majority experience replay buffers D^{min} and D^{maj}, respectively, according to the form of SumTree (Schaul et al. 2015), and obtain the replay memory sets of the minority and majority experience samples $\boldsymbol{E}_{replay}^{min} = \left\{e_i^{min}\right\}_{1 \le i \le d_1} \subset \boldsymbol{E}^{min}$ and $\boldsymbol{E}_{replay}^{maj} = \left\{e_j^{maj}\right\}_{1 \le j \le d_2} \subset \boldsymbol{E}^{maj}$.

Step 2: Calculate the TD errors of the experience samples in $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$, denoted as δ_i^{min} and δ_j^{maj}, respectively, which can be expressed as follows:

$$\delta_i^{min} = \left| y_i^{min} - Q\left(s_i^{min}, a_i^{min}; \theta^C\right) \right|, \quad i = 1, 2, \ldots, d_1, \tag{11}$$

$$\delta_j^{maj} = \left| y_j^{maj} - Q\left(s_j^{maj}, a_j^{maj}; \theta^C\right) \right|, \quad j = 1, 2, \ldots, d_2, \tag{12}$$

where $y_i^{min} = r_i^{min} + \gamma \cdot \max_{a_{i+1}^{min} \in \boldsymbol{A}} Q\left(s_{i+1}^{min}, a_{i+1}^{min}; \theta^T\right)$ and $y_j^{maj} = r_j^{maj} + \gamma \cdot \max_{a_{j+1}^{maj} \in \boldsymbol{A}} Q\left(s_{j+1}^{maj}, a_{j+1}^{maj}; \theta^T\right)$ indicate the target Q-functions in the minority and majority experience samples, respectively; $Q\left(s_i^{min}, a_i^{min}; \theta^C\right)$ and $Q\left(s_j^{maj}, a_j^{maj}; \theta^C\right)$ indicate the current Q-functions in the minority and majority experience samples, respectively; θ^T and θ^C indicate the parameters of the target Q-network and current Q-network, respectively; and d_1 and d_2 indicate the numbers of minority and majority experience samples in $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$, respectively.

Step 3: Calculate the probabilities of minority and majority experience samples selected from $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$ according to the TD error, which are represented as $P\left(e_i^{min}\right)$ and $P\left(e_j^{maj}\right)$, respectively:

$$P\left(e_i^{min}\right) = \frac{\delta_i^{min}}{\sum_1^{d_1} \delta_i^{min}}, \tag{13}$$

$$P\left(e_j^{maj}\right) = \frac{\delta_j^{maj}}{\sum_j^{d_2} \delta_j^{maj}} \tag{14}$$

Step 4: Select k_1 and k_2 ($k_1 = k_2$) minority and majority experience samples from $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$ according to $P\left(e_i^{min}\right)$ and $P\left(e_j^{maj}\right)$, respectively, to construct the stratified prioritized mini-batch.

5.2 Procedure of the DQN-BSPER model

This section describes the introduction of the BSPER technology into the DQN model to construct the DQN-BSPER model. Figure 1 shows the Q-network structure of the DQN-BSPER model. The input layer of the network is the environmental state (the features of

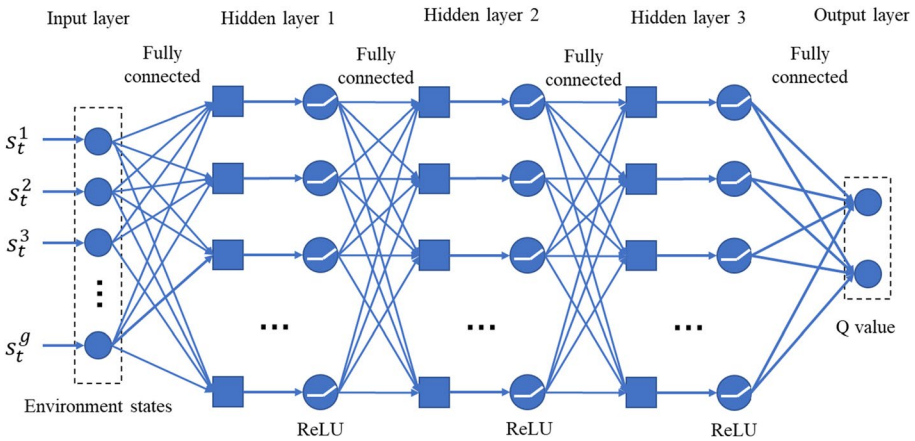


Fig. 1 The Q-network structure of the DQN-BSPER model

the customer sample) $s_t = \left(s_t^1, s_t^2, \ldots, s_t^g\right)$, where the number of nodes in the input layer is the feature number of environment state g, and the output layer is set to two Q-values. According to the Q-value, the customer class label is mapped using a one-hot method, and the nodes in each layer are connected in a fully connected manner.

The core idea of the DQN-BSPER model for CCS is as follows. Firstly, in the training set, the BSPER technology is used to construct the stratified prioritized mini-batch to optimize the loss function, and then the gradient descent algorithm is used to update the Q-network parameters continuously to optimize the classification strategy of the agent in CCS. Finally, in the test set, the trained DQN-BSPER model was used to evaluate customer credit. The detailed modelling steps of the proposed model can be summarised as follows (the modelling process is shown in "Appendix 2").

Step 1: Initialise the parameters of the current Q-network θ_t^C, parameters of the target Q-network θ_t^T, target network update frequency Z, time step t, maximum time step T, episode c, maximum episode C, size of mini-batch k, number of minority experience samples in mini-batch k_1, and number of majority experience samples in mini-batch k_2.

Step 2: Randomly sort the customer samples in the training set and input them into the CCS environment. Each environmental state corresponds to a customer sample; that is, $s_t \leftarrow x_t$.

Step 3: The agent obtains environment state s_t and performs credit scoring actions a_t according to the ε-greedy strategy:

$$\pi\left(a_t|s_t\right) = \begin{cases} 1-\varepsilon, if\ a_t = \underset{a_t \in \boldsymbol{A}}{argmax} Q\left(s_t, a_t; \theta_t^C\right), \\ \varepsilon, else, \end{cases} \tag{15}$$

where the ε-greedy strategy adds randomness to the greedy strategy. That is, the agent selects the action that maximises $Q\left(s_t, a_t; \theta_t^C\right)$ based on the probability $1-\varepsilon$ under environment state s_t; otherwise, an action is randomly selected based on the probability ε. Then, reward r_t is obtained according to the reward function $R\left(a_t, y_t\right)$ (see Eq. (8)), and the environment is transferred to the next state s_{t+1}.

Step 4: According to Step 3, the agent continuously obtains the experience sample $e_t = \left(s_t, a_t, r_t, s_{t+1}\right)$, and obtains $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$ by using the BSPER technology.

Step 5: If $\left|\boldsymbol{E}_{replay}^{min}\right| \le k_1$ or $\left|\boldsymbol{E}_{replay}^{maj}\right| \le k_2$, then go to Step 3 and let $t \leftarrow t+1$; otherwise, use the BSPER technology to construct the stratified prioritized mini-batch. Then, the corresponding loss function of Q-network at the t-th time step $L_{STS}\left(\theta_t^C\right)$ can be expressed as follows:

$$L_{STS}\left(\theta_t^C\right) = \frac{1}{k_1}\sum_{i=1}^{k_1}\left(y_i^{min} - Q\left(s_i^{min}, a_i^{min}; \theta_t^C\right)\right)^2 + \frac{1}{k_2}\sum_{j}^{k_2}\left(y_j^{maj} - Q\left(s_j^{maj}, a_j^{maj}; \theta_t^C\right)\right)^2, \tag{16}$$

$$y_i^{min} = \begin{cases} r_i^{min}, ift = T \\ r_i^{min} + \gamma \cdot \underset{a_{i+1}^{min} \in \boldsymbol{A}}{\max} Q\left(s_{i+1}^{min}, a_{i+1}^{min}; \theta_t^T\right), ift \ne T \end{cases} i = 1, 2, \ldots, k_1 \tag{17}$$

$$y_j^{maj} = \begin{cases} r_j^{maj}, ift = T \\ r_j^{maj} + \gamma \cdot \underset{a_{j+1}^{maj} \in \boldsymbol{A}}{\max} Q\left(s_{j+1}^{maj}, a_{j+1}^{maj}; \theta_t^T\right), ift \ne T \end{cases} j = 1, 2, \ldots, k_2 \tag{18}$$

where y_i^{min} and y_j^{maj} indicate the target Q-functions in the minority and majority experience samples, respectively; $Q\left(s_i^{min}, a_i^{min}; \theta_t^C\right)$ and $Q\left(s_j^{maj}, a_j^{maj}; \theta_t^C\right)$ indicate the current Q-functions in the minority and majority experience samples, respectively; θ_t^T and θ_t^C indicate the parameters of the target and current Q-networks, respectively; k_1 and k_2 ($k_1 + k_2 = k$) indicate the numbers of minority and majority experience samples selected from $\boldsymbol{E}_{replay}^{min}$ and $\boldsymbol{E}_{replay}^{maj}$, respectively; and γ is the discount factor.

Step 6: Use the Adam optimization algorithm to update the parameters of the current Q-network (Schaul et al. 2015):

$$\frac{\nabla L_{STS}\left(\theta_t^C\right)}{\nabla \theta_t^C} = \frac{2}{k_1}\sum_{i=1}^{k_1}\left(y_i^{min} - Q\left(s_i^{min}, a_i^{min}; \theta_t^C\right)\right)\frac{\nabla L_{STS}\left(Q\left(s_i^{min}, a_i^{min}; \theta_t^C\right)\right)}{\nabla \theta_t^C}$$

$$+\frac{2}{k_2}\sum_{j=1}^{k_2}\left(y_j^{maj} - Q\left(s_j^{maj}, a_j^{maj}; \theta_t^C\right)\right)\frac{\nabla L_{STS}\left(Q\left(s_j^{maj}, a_j^{maj}; \theta_t^C\right)\right)}{\nabla \theta_t^C}, \tag{19}$$

$$m_t \leftarrow \beta_1 m_t + \left(1-\beta_1\right)\frac{\nabla L_{STS}\left(\theta_t^C\right)}{\nabla \theta_t^C}, \tag{20}$$

$$d_t \leftarrow \beta_2 d_t + \left(1-\beta_2\right)\left(\frac{\nabla L_{STS}\left(\theta_t^C\right)}{\nabla \theta_t^C}\right)^2, \tag{21}$$

$$\widehat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}, \tag{22}$$

$$\widehat{d}_t \leftarrow \frac{d_t}{1-\beta_2^t}, \tag{23}$$

$$\theta_t^C \leftarrow \theta_t^C - \frac{\alpha}{\sqrt{\widehat{d}_t} + \epsilon}\widehat{m}_t, \tag{24}$$

where $\frac{\nabla L_{STS}\left(\theta_t^C\right)}{\nabla \theta_t^C}$ indicates the gradient of $L_{STS}\left(\theta_t^C\right)$ at the tth time step; m_t and d_t indicate the first- and second-order moments of $\frac{\nabla L_{STS}\left(\theta_t^C\right)}{\nabla \theta_t^C}$, respectively; β_1 and β_2 indicate the exponential decay rates of the first- and second-order moments, respectively; $\widehat{m}_t$ and $\widehat{d}_t$ indicate the deviation correction values of m_t and d_t, respectively; Eq. (24) indicates that the parameters of the Q-network θ_t^C are updated by combining Eqs. (22) and (23); α indicates the learning rate of the current Q-network; and ϵ indicates a constant value to prevent $\sqrt{\widehat{d}_t} + \epsilon$ from generating the 0 value. Then, for each Z time step, the parameters of the current Q-network θ_t^C are assigned to the parameters of the target Q-network θ_t^T; that is, let $\theta_{Z|t}^T \leftarrow \theta_{Z|t}^C$.

Step 7: If $t \le T$, then proceed to Step 3 and let $t \leftarrow t+1$; otherwise, proceed to Step 8.

Step 8: If $c \leq C$, then go to Step 2 and let $c \leftarrow c + 1$; otherwise, stop training, output the trained current Q-network, and use it to conduct CCS according to the greedy strategy in the test set.

The current Q-network obtained through the above steps is a parameterised optimal Q-function; therefore, the strategy corresponding to the network is also the optimal classification strategy. The pseudo-codes of the CCS environment simulation and DQN-BSPER model are shown in "Appendix 3".

6 Experimental design

This section introduces the experimental design. Firstly, two CCS datasets in P2P lending and preprocessing are described, the detailed experimental process and main parameter settings of the models are discussed in detail, and four EMs are introduced to evaluate the performance of the models.

6.1 Data set description and preprocessing

To analyse the CCS performance of the DQN-BSPER model, we selected two CCS datasets in P2P lending (IFCD and Leading Club). The IFCD dataset was obtained from an Internet financial company in China. The original dataset contained 1110 features. The Leading Club dataset was acquired from the first quarter data of Lending Club, which is a P2P lending platform in the United States in 2016, and the original dataset contained 110 features. According to the regulations of the Basel Banking Regulatory Association, we labelled customers whose loans were overdue for more than 90 days as having bad credit and others as having good credit.

The two CCS datasets in P2P lending contained redundant features and missing values, which were processed as follows. First, we deleted some meaningless features through observation, such as loan number, zip code, and customer ID. Then, the features with missing rates of more than 30% were eliminated. Finally, the recursive feature elimination method Wrapper was used to eliminate the features with the minimum absolute weights continuously. The basic information about the two CCS datasets in P2P lending after preprocessing is shown in Table 1, including the dataset name (the abbreviation in parentheses), number of features, number of customer samples, number of customer samples with good credit, number of customer samples with bad credit, and imbalanced ratio (*IR*), which is defined as the proportion of the number of the majority samples (customers with good credit) relative to the number of minority samples (customers with bad credit). Obviously, the greater the *IR*, the higher the class distribution imbalance. As shown in Table 1, the two CCS datasets are imbalanced.

Table 1 The basic information of two CCS data sets in P2P lending after preprocessing

Data sets	Features	Samples	Good credit	Bad credit	*IR*
IFCD (IF)	50	23,319	21,583	1736	12.43
Leading Club (LE)	90	133,887	109,888	23,999	4.58

6.2 Experimental procedure and parameter settings

In this study, we performed fivefold cross validation in each dataset to obtain the calculation results for the models. Firstly, each dataset was divided equally into five subsets at random. In each experiment, one subset was used as the test set, $\boldsymbol{D}_{test}$, and the remaining subsets were used as the training set, $\boldsymbol{D}_{train}$. Then, the DQN-BSPER model and other models were trained in $\boldsymbol{D}_{train}$. Finally, the trained models were used to classify the samples in $\boldsymbol{D}_{test}$. This process was repeated five times to ensure that each subset was used once. The entire process constitutes fivefold cross validation. To obtain more stable and reliable experimental results, we repeated the fivefold cross validation 10 times and took the average value as the final calculation result. In addition, we recorded the network loss each time the model traversed the training set in the process of fivefold cross validation and averaged the training network loss at the end of training. All experiments in this study were run on Python 3.6, in a Windows 10×64 bit system equipped with an Intel (R) Core (TM) i9 processor.

We mainly referred to previous studies (Lin et al. 2020; Liu et al. 2020; Mnih et al. 2015) to set the parameters of the DRL models. "Appendix 4" shows the main parameter settings of the DQN-BSPER and other DRL models. In particular, the size of the replay memory buffers of the DQN-BSPER and DQN-SER models D^{min} and D^{maj} was 500. In addition, the current and target Q-networks of the DQN-BSPER model had the same structure. To ensure the fairness of the experiment, we adjusted the discount factors of the DQN, DQN-SER, DQN-PER, and DQN-IRF models many times and determined their optimal values. Specifically, the optimal discount factor of the DQN-SER model was 0.1, and the optimal discount factor of the DQN, DQN-PER, and DQN-IRF models was 0.3. In all experiments, the network parameters of the five DRL models were equal. In addition, we used the grid search and fivefold cross validation (Gunnarsson et al. 2021) to ensure the excellent performance of the other classification models.

6.3 Evaluation measures

The confusion matrix method can intuitively evaluate classification models performance (Batista et al. 2004). Table 2 shows the confusion matrix of CCS, from which many different EMs can be obtained.

$$True\ positive\ rate\ (TPR) = \frac{TP}{TP + FN}, \tag{25}$$

$$True\ negative\ rate\ (TNR) = \frac{TN}{TN + FP}, \tag{26}$$

Table 2 The confusion matrix of CCS

	Predicted positive	Predicted negative	Total
Actual positive (bad credit customer)	TP	FN	TP + FN
Actual negative (good credit customer)	FP	TN	FP + TN
Total	TP + FP	FN + TN	TP + FN + FP + TN

$$False\ positive\ rate\ (FPR) = \frac{FP}{TN + FP}, \tag{27}$$

$$Precision = \frac{TP}{TP + FP}, \tag{28}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{29}$$

However, there are some contradictions between these EMs, so some new comprehensive measures have been proposed, such as $F1$ (Tang et al. 2008) and AUC (Bradley 1997). $F1$ is a combination of precision and recall:

$$F1 = \frac{(1 + \beta^2) \times recall \times precision}{\beta^2 \times recall + precision}, \tag{30}$$

where $recall = TPR$ and β is the relative coefficient of *recall* to *precision*. In this study, we design $\beta = 1$.

In this research, we used the approximate representation of the AUC for the binary classification problem (Loyola-González et al. 2016):

$$AUC = \frac{TPR + TNR}{2}. \tag{31}$$

Therefore, we used four TPR, ACC, $F1$, and AUC in this study. The larger the values of them, the better the performance of the CCS model.

7 Experimental results and analysis

We empirically analysed the CCS performance of the DQN-BSPER model using two real-world P2P lending CCS datasets. Firstly, we assessed the effects of parameters γ and k_1/k_2 on the CCS performance of the DQN-BSPER model. Subsequently, the CCS performance of DQN-BSPER and the four benchmark DRL models DQN (Mnih et al. 2013), DQN-SER (Chen et al. 2018), DQN-PER (Schaul et al. 2015), and DQN-IRF (Lin et al. 2020) were statistically compared, and their convergence performance was compared. Next, the CCS performance of the DQN-BSPER model and seven traditional benchmark classification models was statistically compared. Finally, the impact of the imbalanced class distribution on the CCS performance of the DQN-BSPER model was analysed.

7.1 Parameter sensitivity analysis

To analyse the effects of the discount factor γ and the proportion of minority and majority experience samples in the stratified prioritized mini-batch k_1/k_2 on the CCS performance of the DQN-BSPER model, we set 11 different types ofγ; that is, we let$\gamma = \{0, 0.1, 0.2, \ldots, 1\}$. A large difference between k_1 and k_2 may significantly affect the DQN-BSPER model performance. Therefore, we set 11 different values ofk_1/k_2: $\{6/1, 5/1, 4/1, \ldots, 1/6\}$. Owing

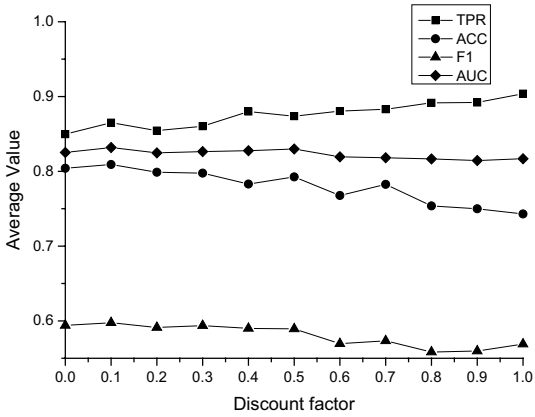


Fig. 2 The average values of EMs of the DQN-BSPER model under 11 values of γ

to the lack of prior experience, we temporarily set $k_1/k_2 = 1$ when analysing the effects of the discount factor on the DQN-BSPER model performance.

Figure 2 depicts the average values of the EMs of the DQN-BSPER model under 11 values of γ (the average value of each EM is the average value of the classification results of the model on two CCS datasets in P2P lending). "Appendix 5" shows the CCS performance of the DQN-BSPER model under the 11 discount factors. The bold values indicate the best CCS performance in each column, the numbers in parentheses indicate the rankings of the DQN-BSPER model with different discount factors, and the last column indicates the average ranking value of the DQN-BSPER model performance with each discount factor. The smaller the ranking value, the better the model performance. As shown in Fig. 2, with respect to *TPR*, the CCS performance of the DQN-BSPER model shows a gradual upward trend with increasing γ. According to *ACC*, *F*1, and *AUC*, the DQN-BSPER model performance shows a gradual downward trend with increasing γ. To determine the most appropriate discount factor, we ranked the DQN-BSPER model performance under different discount factors. As can be seen in "Appendix 5", the DQN-BSPER model has the smallest ranking value when $\gamma = 0.1$. The DQN-BSPER model has the largest ranking value when $\gamma = 0.8$.

Figure 3 presents the average values of the EMs of the DQN-BSPER model under 11 values of k_1/k_2. Furthermore, we ranked the CCS performance of the DQN-BSPER

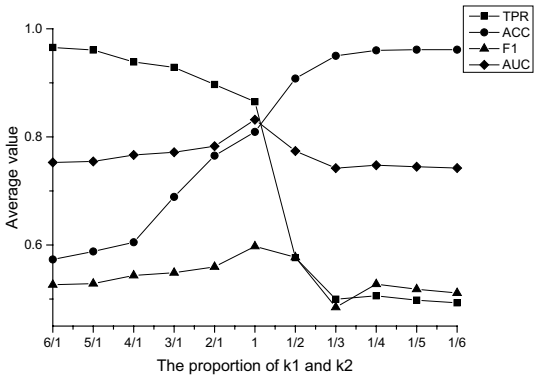


Fig. 3 The average values of EMs of the DQN-BSPER model under 11 values of k_1/k_2

model under different k_1/k_2 values, and the results are shown in "Appendix 6". As depicted in Fig. 3, with respect to *TPR*, the DQN-BSPER model performance shows a gradual downward trend with increasing k_1/k_2. With respect to *ACC*, the DQN-BSPER model performance shows a gradual upward trend with increasing k_1/k_2. With respect to $F1$ and *AUC*, the DQN-BSPER model performance firstly shows an upward trend and then a downward trend. As shown in "Appendix 6", the DQN-BSPER model has the smallest ranking value when $k_1/k_2 = 1$.

It can be seen from the experimental results that the CCS performance of the DQN-BSPER model is the best when $\gamma = 0.1$ and $k_1/k_2 = 1$. Interestingly, with increasing γ, the *TPR* gradually increases, whereas *ACC*, $F1$, and *AUC* gradually decrease. This observation shows that the greater γ, the more majority samples will be wrongly divided into the minority by the DQN-BSPER model, resulting in poor overall performance of the model. When $\gamma = 0.1$, the DQN-BSPER model performance is the best, which also demonstrates that there is a weak sequence correlation between the samples in the CCS dataset, whereas identifying the correlation may make the DQN-BSPER model more robust (Lei et al. 2020). In addition, setting k_1/k_2 too high or too low leads to poor CCS performance of the DQN-BSPER model. When $k_1/k_2 = 1$, the DQN-BSPER model has the highest $F1$ and *ACC* values, which verifies that balancing the class distribution of different experience samples in the stratified prioritized mini-batch can improve the CCS performance of the DQN-BSPER model the most effectively.

7.2 Credit scoring performance comparison of DQN-BSPER and four other benchmark DRL models

This section compares the CCS performance of DQN-BSPER and the other four benchmark DRL models, DQN, DQN-SER, DQN-PER, and DQN-IRF, on the IF and LE datasets according to four EMs. Table 3 presents the CCS performance of DQN-BSPER and the other four benchmark DRL models on the IF and LE datasets. The bold font indicates the model with the best CCS performance in each column, and the numbers in parentheses indicate the model performance rankings. The last column indicates the average ranking of the performance of each model.

Table 3 demonstrates that the DQN-BSPER model has the smallest average ranking, indicating that the model has the best CCS performance. To analyse whether there are statistically significant differences in CCS performance among DQN-BSPER and the other four benchmark DRL models, we used the nonparametric Wilcoxon rank sum test

Table 3 The CCS performance of DQN-BSPER and four other benchmark DRL models on the IF and LE datasets

Model	IF dataset				LE dataset				Average rank
	TPR	*ACC*	*F1*	*AUC*	*TPR*	*ACC*	*F1*	*AUC*	
DQN-BSPER	**0.7464**$^{(1)}$	**0.6296**$^{(1)}$	**0.2260**$^{(1)}$	**0.6779**$^{(1)}$	**0.9838**$^{(1)}$	0.9888$^{(2)}$	**0.9690**$^{(1)}$	**0.9857**$^{(1)}$	**1.13**
DQN	0.0191$^{(4)}$	0.6284$^{(2)}$	0.0289$^{(4)}$	0.5070$^{(4)}$	0.9585$^{(4)}$	0.9819$^{(5)}$	0.9673$^{(5)}$	0.9827$^{(3)}$	3.88
DQN-SER	0.6887$^{(2)}$	0.6202$^{(4)}$	0.2211$^{(2)}$	0.6680$^{(2)}$	0.9829$^{(2)}$	0.9852$^{(3)}$	0.9683$^{(2)}$	0.9836$^{(2)}$	2.38
DQN-PER	0.0422$^{(3)}$	0.6241$^{(3)}$	0.0782$^{(3)}$	0.5182$^{(3)}$	0.9635$^{(3)}$	0.9821$^{(4)}$	0.9677$^{(3)}$	0.9809$^{(4)}$	3.25
DQN-IRF	0.0163$^{(5)}$	0.6107$^{(5)}$	0.0247$^{(5)}$	0.5036$^{(5)}$	0.9499$^{(5)}$	**0.9908**$^{(1)}$	0.9674$^{(4)}$	0.9748$^{(5)}$	4.36

The bold indicates the best ranked model in each column

Table 4 Wilcoxon rank sum test results for DQN-BSPER and four other benchmark DRL models on the IF and LE datasets

Comparison	$T=\text{Min}(R^+, R^-)$	CV	p-value	Hypothesis
DQN-BSPER vs. DQN	Min(36, 0) = 0	3	0.012	Reject
DQN-BSPER vs. DQN-PER	Min(36, 0) = 0	3	0.012	Reject
DQN-BSPER vs. DQN-SER	Min(36, 0) = 0	3	0.012	Reject
DQN-BSPER vs. DQN-IRF	Min(34, 2) = 2	3	0.025	Reject

R^+ indicates the sum of the rankings of all cases in which the performance of the former is better than that of the latter when comparing two models; R^- indicates the sum of the rankings of all cases in which the performance of the former is worse than that of the latter

(Wilcoxon 1992). The null hypothesis is that the two comparative models have the same CCS performance. In this paper, we set the significance level $\alpha = 0.05$. Then, at the 95% confidence level, when the statistical amount is 8 ($= 2 \times 4$), the critical value (CV) is 3. The Wilcoxon rank sum test results of DQN-BSPER and four other benchmark models are provided in Table 4. In Table 4, if $T = \text{Min}(R^+, R^-) \leq 3$, the null hypothesis is rejected. In particular, if $T = R^-$ and $T \leq 3$, it means that the former performance is statistically significantly better than the latter. If $T = R^+$ and $T \leq 3$, then the opposite is true.

The test results in Table 4 indicate that when comparing the DQN-BSPER model with the DQN, DQN-SER, DQN-PER, and DQN-IRF models, R^+ is less than the CV; in other words, at the 95% confidence level, the CCS performance of the DQN-BSPER model is statistically significantly better than those of the other four benchmark models. More importantly, the CCS performances of the DQN-SER and DQN-PER models are better than that of the DQN model, whereas the DQN-BSPER model is superior to the DQN-SER and DQN-PER models. Thus, although the CCS performance of the DQN model can be improved by using SER or PER technology alone, the effect of improvement is very limited, whereas the combination of SER and PER technologies can further improve the performance of the DQN model. In addition, for the IF dataset, the CCS performance of the DQN-IRF model is very poor. This poor performance may occur because the DQN-IRF

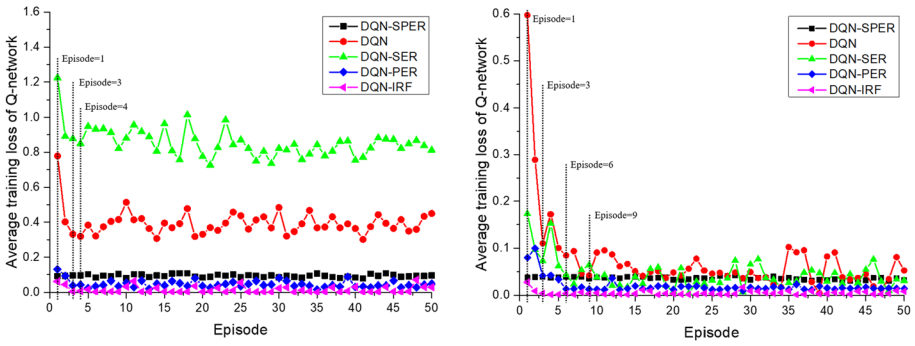


Fig. 4 The average training losses of the Q-networks of DQN-BSPER and other four benchmark DRL models on two CCS training sets (the left figure shows the results on the IF data set and the right figure shows the results on the LE dataset)

model (Lin et al. 2020) is mainly applied to classification tasks with unstructured data (image and text data), whereas the CCS data are structured, which may lead to the failure of the reward function, affecting the CCS performance of the DQN-IRF model.

7.3 Convergence performance comparison of DQN-BSPER and four other benchmark DRL models

To analyse the effects of the SPER technology on the convergence performance of the DQN model, this section compares the convergence performances of the DQN-BSPER, DQN, DQN-SER, DQN-PER, and DQN-IRF models according to four EMs on the IF and LE datasets. Figure 4 shows the average training loss of the Q-networks of DQN-BSPER and the other four benchmark DRL models on two CCS training sets.

Figure 4a shows the average training losses of the Q-networks of the five DQN models on the IF dataset. The average training loss of the Q-network of the DQN-BSPER model reaches a stable value when episode = 1, indicating that the DQN-BSPER model converges when episode = 1, and the value fluctuates slightly as the number of episodes increases. The average training loss of the Q-network of the DQN model is very large when episode = 1; then, the value decreases rapidly and reaches the local minimum when episode = 4, but the value fluctuates strongly as the number of episodes increases. The average training loss of the Q-network of the DQN-SER model is larger than those of the other models, and the fluctuation is also very strong. The average training losses of the Q-networks of DQN-PER and DQN-IRF models reach stable values when episode = 3; that is, the DQN-PER and DQN-IRF models reach a convergence state when episode = 3. Figure 4b shows the average training losses of the Q-networks of the five DQN models on the LE dataset. The average training loss of the Q-network of the DQN-BSPER model reaches a stable value when episode = 1, indicating that the DQN-BSPER model converges when episode = 1, and the value fluctuates slightly as the number of episodes increases. The average training loss of the Q-network of the DQN model is very large when episode = 1 and reaches the local minimum when episode = 9, but the value still fluctuates strongly as the number of episodes increases. The average training loss of the Q-network of the DQN-SER model also fluctuates significantly. The average training losses of the Q-networks of the DQN-PER and DQN-IRF models reach stable values when episode = 6 and episode = 3, respectively; that is, the DQN-PER and DQN-IRF models reach a convergence state when episode = 6 and episode = 3, respectively, and their average training losses of the Q-network are relatively small.

The experimental results show that the Q-network of the DQN-BSPER model has the fastest convergence speed and that the average training loss of the Q-network is the most stable. Thus, the SPER technology can effectively improve the convergence performance of the DQN model; that is, it can not only accelerate the convergence speed of the Q-network, but also make the Q-network more stable. In addition, the average training losses of the Q-networks of the DQN-PER and DQN-IRF models are very small, but the results in Sect. 7.2 show that their CCS performance is relatively poor, which indicates that over-fitting may occur. Interestingly, the fluctuation in the average training losses of the DQN and DQN-SER models is stronger than that of the DQN-BSPER and DQN-PER models. The main reason may be that the DQN-BSPER and DQN-PER models consider the priority of the experience samples when building the mini-batch. Thus, PER technology can improve the stability of the DQN model more effectively than SER technology.

Table 5 Friedman and Iman–Davenport test results for DQN-BSPER and seven traditional benchmark classification models

	Test value	Distribution value	Hypothesis
Friedman	98.58	14.07	Reject
Iman–Davenport	32.66	2.07	Reject

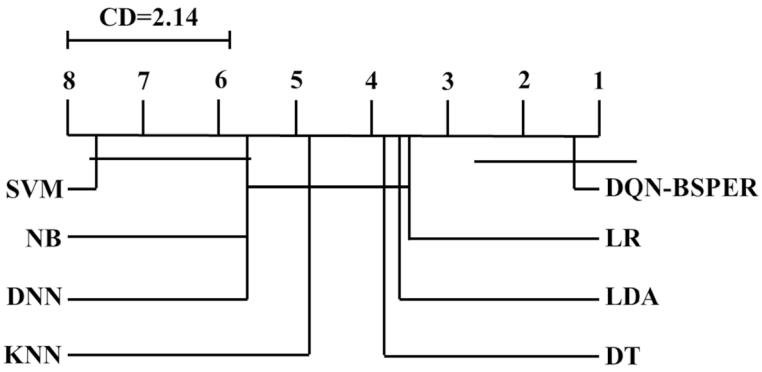


Fig. 5 Nemenyi post hoc test results for DQN-BSPER and seven traditional benchmark classification models

7.4 Credit scoring performance comparison of DQN-BSPER and seven traditional classification models

This section compares the CCS performance of DQN-BSPER and seven traditional benchmark classification models on the IF and LE datasets. The results are presented in "Appendix 7", in which the bold font indicates the classification model with the best CCS performance in each line, and the numbers in parentheses indicate the performance rankings of the classification models. The last line indicates the average ranking of each classification model. To analyse whether there are statistically significant differences between DQN-BSPER and the seven traditional benchmark classification models, we used the non-parametric statistical test method proposed by (Demšar 2006), namely, the Friedman test (Friedman 1940) and Iman–Davenport test (Iman and Davenport 1980). If there were statistically significant differences, we further used the Nemenyi post hoc test method to compare the eight classification models. In addition, in order to achieve fair results, we used the resampling methods ROS and RUS to balance the class distribution of the two training sets to form four new training sets (the class distribution ratio of the balanced training set was 1:1) during each training process. Each dataset corresponded to two new cases; that is, the IF dataset corresponded to IF (ROS) and IF (RUS), whereas the LE dataset corresponded to LE (ROS) and LE (RUS).

The results in "Appendix 7" show that the DQN-BSPER model has the smallest average ranking, indicating that it achieves the best CCS performance. We used the Friedman and Iman–Davenport tests to analyse further whether there were statistically significant differences between DQN-BSPER and the seven traditional benchmark classification models. The null hypothesis of the two test methods was that the CCS performance of the eight classification models were the same. We used the χ^2 distribution with 7 freedom degree and the F distribution with 7 and 161 ($= 7 \times 23$) freedom degree. Table 5 list the test

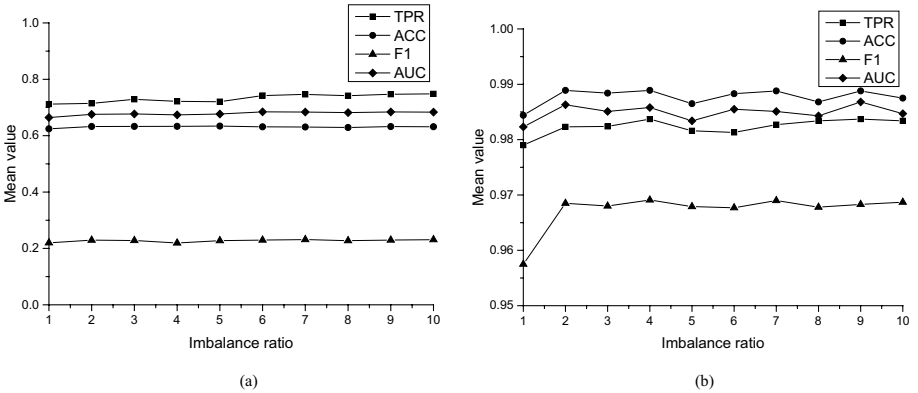


Fig. 6 The CCS performance of the DQN-BSPER model under 10 imbalance ratios for the **a** IF and **b** LE datasets

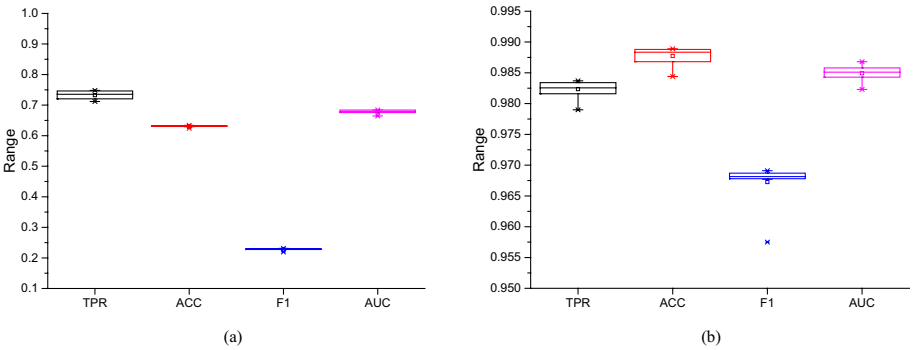


Fig. 7 Boxplots of four EMs of the DQN-BSPER model for the **a** IF and **b** LE datasets

results. If the test value was greater than the distribution value, the null hypothesis was rejected. Therefore, according to results of Table 5, we rejected the null hypothesis and concluded that there were statistically significant differences among the eight classification models in the CCS performance at the 95% confidence level.

After the null hypothesis was rejected, we used the Nemenyi post hoc test. The judgement rule of this test is that if the difference between the average values of any two classification models is greater than the critical difference (CD), then the null hypothesis that the two classification models performance is the same at the 95% confidence level is that there are significant differences in the CCS performance between the two classification models. When the number of classification models is 8 and the CV (Demšar 2006) is 3.03, the corresponding critical distance $\mathrm{CD} = 3.03\sqrt{(8*9)/(6*24)} \approx 2.14$. The test results are presented in Fig. 5. If there are segments connected between the two classification models, then there are no statistically significant differences. The results in Fig. 5 show that the CCS performance of the DQN-BSPER model is statistically significantly better than those of the other seven traditional benchmark classification models at the 95% confidence level.

7.5 Impact of imbalanced class distribution on the customer credit scoring performance of the DQN-BSPER model

To analyse the effects of imbalanced class distribution on the CCS performance of the DQN-BSPER model, we set 10 imbalance ratios for the IF and LE datasets. Specifically, we firstly randomly sampled from the majority samples in the training set according to the multiple of the number of minority samples and then combined the sampled majority samples with all minority samples to form a new training set. According to this method, training sets with 10 imbalance ratios ($IR = \{1, 2, \ldots, 10\}$) were obtained, and the DQN-BSPER model performance was analysed according to four EMs. Figure 6 shows the CCS performance of the DQN-BSPER model under 10 imbalanced ratios on the IF and LE datasets. Furthermore, we used boxplots to analyse the dispersion degrees of the four EM values, as shown in Fig. 7.

As can be seen from Fig. 6a, under 10 types of *IR*, the values of *TPR*, *ACC*, and *AUC* of the DQN-BSPER model are higher than the *F*1 values. In Fig. 6b, under 10 types of *IR*, the values of *TPR*, *ACC*, and *AUC* of the DQN-BSPER model are also higher than the *F*1 values. When $IR = 1$, the four evaluation values of the DQN-BSPER model are relatively low. In Fig. 7a, the box corresponding to the *TPR* is the longest, whereas the *ACC*, *AUC*, and *F*1 boxes are relatively short. In Fig. 7b, the box for *F*1 is the shortest, whereas the *TPR*, *ACC*, and *AUC* boxes are relatively long. It can be seen from the experimental results that for the IF and LE datasets, the boxes of the four EM values of the DQN-BSPER model are very narrow, which demonstrates that their dispersion degree is quite small. Thus, the fluctuations in the *TPR*, *ACC*, *F*1, and *AUC* values of the DQN-BSPER model are very weak under different imbalance ratios. This finding indicates that the imbalanced class distribution has a limited effect on the CCS performance of the DQN-BSPER model.

8 Conclusions and future works

This paper proposed DRL based on a stratified prioritized experience replay model, that is, the DQN-BSPER model. The model optimizes the loss function by combining SER and PER to improve its CCS performance. Further, we introduced four EMs to conduct an empirical analysis of two real-world CCS datasets in P2P lending. In the parameter analysis experiment, we found that the CCS performance of the DQN-BSPER model was the best when $\gamma = 0.1$ and $k_1/k_2 = 1$. This result shows that there is a relatively weak sequence correlation between the CCS dataset samples. The balanced stratified prioritized mini-batch is more conducive to improving the CCS performance of the DQN-BSPER model. In the second experiment, we found that the performance of the DQN-BSPER model was statistically significantly better than those of the DQN, DQN-SER, DQN-PER, and DQN-IRF models. By analysing the convergence performance of the DQN-BSPER model, we found that SPER technology could not only accelerate the convergence speed of the Q-network, but also improve the stability of the network. In addition, in the comparing experiment, we found that the DQN-BSPER model performance was statistically significantly better than those of the other seven traditional benchmark classification models. Interestingly, in the experiment in which the effects of the imbalanced class distribution on the DQN-BSPER model performance were investigated, we found that the effect of the imbalanced class distribution on the CCS performance of the DQN-BSPER model was very limited.

The following aspects will be considered in future work. First, we will introduce decision making methods (Kou et al. 2021a, b, 2021a; Kou et al. 2014; Wang et al. 2021) into

the reward function to improve DRL classification performance. Second, we will attempt to use a network structure identifying feature importance (Xiao et al. 2020) to improve the interpretability of the deep reinforcement learning model, which is of great significance for enterprise management. Finally, we will combine relationship network (Kou et al. 2021a, b) with DRL to develop a multi-agent classification system.

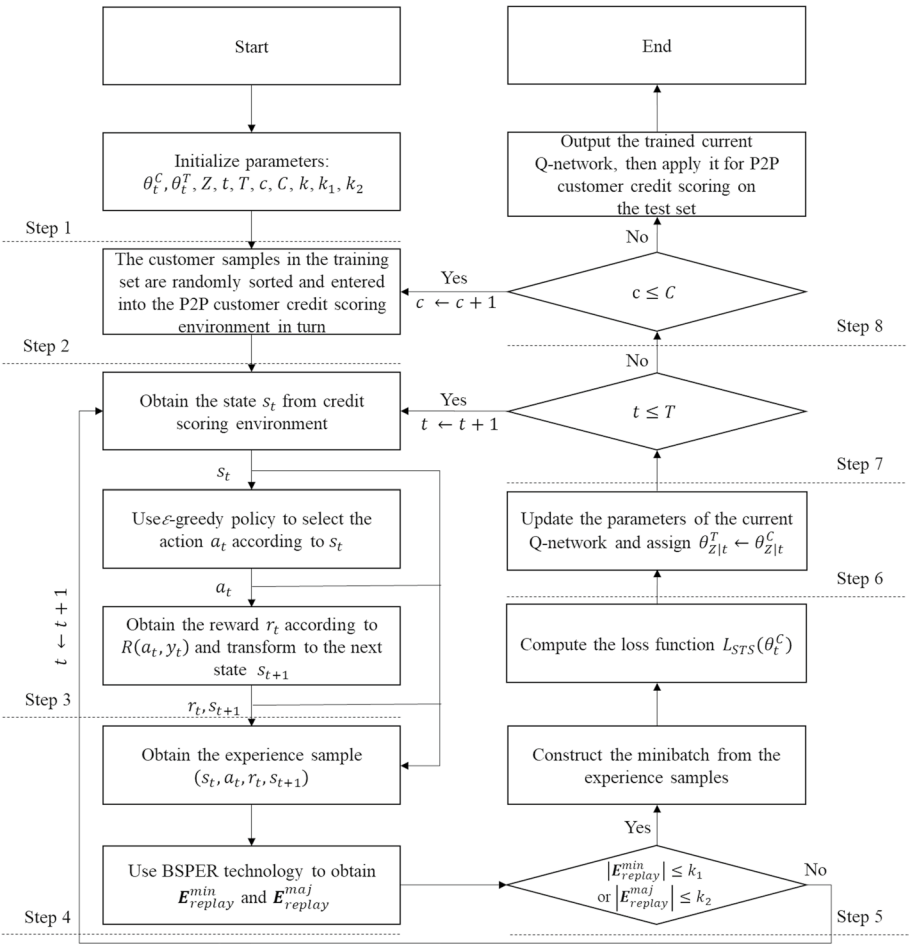
Appendices

Appendix 1: Notations and definitions

Notations	Definitions
$\boldsymbol{D}_{train}$	The customer credit scoring training set
$\boldsymbol{D}_{test}$	The customer credit scoring testing set
$\boldsymbol{D}_{min}$	The minority sample set
$\boldsymbol{D}_{maj}$	The majority sample set
x_t	The tth customer sample
y_t	The class label of the tth customer sample
$\boldsymbol{S}$	The state space of the environment
$\boldsymbol{A}$	The action space of the agent
s_t	The environment state at the tth time step
a_t	The action of the agent at the tth time step
r_t	The reward obtained by the agent according to the reward function R at the tth time step
γ	The discount factor
R_t	The cumulative reward obtained by the agent from time step t to terminal step in each episode
e_t	The experience sample at the tth time step
$\boldsymbol{E}_t$	The experience sample set at the tth time step
$\boldsymbol{E}^{min}$	The minority experience sample set
$\boldsymbol{E}^{maj}$	The majority experience sample set
D	The replay memory buffer for storing experience samples
D^{min}	The replay memory buffer for storing minority experience samples
D^{maj}	The replay memory buffer for storing majority experience samples
$\boldsymbol{E}^{min}_{replay}$	The experience sample set stored in D^{min}
$\boldsymbol{E}^{maj}_{replay}$	The experience sample set stored in D^{maj}
d_1	The number of experience samples in $\boldsymbol{E}^{min}_{replay}$
d_2	The number of experience samples in $\boldsymbol{E}^{maj}_{replay}$
e^{min}_t	The minority experience sample at the tth time step
e^{maj}_t	The majority experience sample at the tth time step
θ^C_t	The parameters of the current Q-network at the tth time step
θ^T_t	The parameters of the target Q-network at the tth time step
k	The size of the mini-batch
k_1	The number of experience samples sampled from $\boldsymbol{E}^{min}_{replay}$
k_2	The number of experience samples sampled from $\boldsymbol{E}^{maj}_{replay}$

Notations	Definitions
Z	The updating frequency of the target Q-network
T	The maximum time step
C	The maximum episode
α	The learning rate of the current Q-network
TP	The number of positive samples predicted as positive
TN	The number of negative samples predicted as negative
FP	The number of negative samples predicted as positive
FN	The number of positive samples predicted as negative

Appendix 2: The process of using the DQN-BSPER model for customer credit scoring in P2P lending



.

Appendix 3: The pseudo-codes

Algorithm 1 shows the pseudo-code of CCS environment simulation.

Input: a_t, y_t

Output: r_t

1: Reward function $R(a_t, y_t)$:

2: if $a_t = y_t$:

3: $r_t = 1$;

4: if $a_t \neq y_t$:

5: $r_t = -1$;

8: end if

Algorithm 2 shows the pseudo-code of the DQN-BSPER model for CCS in P2P lending.

Input: The training set $\boldsymbol{D}_{train} = \{(x_t, y_t)\}_{1 \le t \le T}$, the test set $\boldsymbol{D}_{test} = \{x'_u\}_{1 \le u \le U}$;

Output: The credit scoring action set $\{a'_u\}_{1 \le u \le U}$ in $\boldsymbol{D}_{test}$.

Training process:

Initialize: the parameters of the current Q-network θ_t^C, the parameters of target Q-network θ_t^T, the updating frequency of the target Q-network Z, the time step t, the maximum time step T, the episode c, the maximum episode C, the number of minority experience samples in the mini-batch k_1, and the number of majority experience samples in the mini-batch k_2.

1: For episode c = 1 to C do:

2: Shuffle the training set $\boldsymbol{D}_{train}$;

3: Initialize environment state $\boldsymbol{S} = \{s_t\}_{1 \le t \le T} \leftarrow \{x_t\}_{1 \le t \le T}$

4: For time step $t = 1$ to T do:

5: Select an action based ε-greedy policy:

6: Probability ε randomly select a credit scoring action $a_t \in A$

7: Probability $(1 - \varepsilon)$ select a credit scoring action $a_t = arg \max_{a_t \in \boldsymbol{A}} Q(s_t, a_t; \theta_t^C)$

8: Obtain reward $r_t = R(a_t, y_t)$

9: If s_t is a minority sample:

10: Store the experience sample in D^{min} to obtain $\boldsymbol{E}_{replay}^{min}$

11: else:

12: Store the experience sample in D^{maj} to obtain $\boldsymbol{E}_{replay}^{maj}$

13: If $\left|\boldsymbol{E}_{replay}^{min}\right| > k_1$ and $\left|\boldsymbol{E}_{replay}^{maj}\right| > k_2$:

14: Use SPER technology to construct the stratified prioritized mini-batch

15: Calculate loss function $L_{STS}(\theta_t^C)$ based on Eqs. (16) - (18)

16: Update θ_t^C based on Eqs. (19) - (24)

17: Update the target Q-network each Z time steps: $\theta_{Z|t}^T \leftarrow \theta_{Z|t}^C$

18: else:

19: End if

20: End for

21: End for

22: Output the parameters of the trained current Q-network $\theta_{trained}^C$

Testing process:

23: Shuffle the test set $\boldsymbol{D}_{test}$;

24: Initialize the sequence state $\boldsymbol{S}' = \{s'_u\}_{1 \le u \le U} \leftarrow \{x'_u\}_{1 \le u \le U}$

25: For time step $u = 1$ to U do:

26: $a'_u = arg \max_{a'_u \in \boldsymbol{A}} Q(s'_u, a'_u; \theta_{trained}^C)$

27: End for

Appendix 4: The main parameters setting of DQN-BSPER and other reinforcement learning models

DRL models	Parameters setting
DQN-BSPER	Initial exploration value = 0.1, final exploration value = 0.01, exploration attenuation = 10,000, $\left\|D^{min}\right\| = \left\|D^{maj}\right\| = 500$, $C = 50$, $Z = 500$, $k = 128$, $\alpha = 0.00025$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 1e-8$, hidden layers of the neural network = 3, number of neurons in the hidden layer = 50, activation function in hidden layers = ReLU
DQN-SER	Initial exploration value = 0.1, final exploration value = 0.01, exploration attenuation = 10,000, $\gamma = 0.1$, $\left\|D^{min}\right\| = \left\|D^{maj}\right\| = 500$, $C = 50$, $Z = 500$, $k = 128$, $\alpha = 0.00025$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 1e-8$, hidden layers of the neural network = 3, number of neurons in the hidden layer = 50, activation function in hidden layers = ReLU
DQN, DQN-PER, DQN-IRF	Initial exploration value = 0.1, final exploration value = 0.01, exploration attenuation = 10,000, $\gamma = 0.3$, $\|D\| = 1000$, $C = 50$, $Z = 500$, $k = 128$, $\alpha = 0.00025$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\tau = 1e-8$, hidden layers of the neural network = 3, number of neurons in the hidden layer = 50, activation function in hidden layers = ReLU

Appendix 5: The CCS performance of the DQN-BSPER model under 11 kinds of discount factors

	IF dataset				LE dataset				Average rank
γ	*TPR*	*ACC*	*F1*	*AUC*	*TPR*	*ACC*	*F1*	*AUC*	
0	$0.7176^{(11)}$	$0.6196^{(2)}$	$0.2192^{(4)}$	$0.6646^{(7)}$	$0.9817^{(8)}$	$0.9886^{(2)}$	$0.9686^{(3)}$	$0.9859^{(2)}$	4.88
0.1	$0.7464^{(8)}$	$\mathbf{0.6296^{(1)}}$	$\mathbf{0.2260^{(1)}}$	$\mathbf{0.6779^{(1)}}$	$0.9838^{(4)}$	$\mathbf{0.9888^{(1)}}$	$\mathbf{0.9690^{(1)}}$	$0.9857^{(3)}$	**2.50**
0.2	$0.7291^{(10)}$	$0.6106^{(3)}$	$0.2179^{(6)}$	$0.6651^{(6)}$	$0.9798^{(11)}$	$0.9871^{(5)}$	$0.9645^{(5)}$	$0.9842^{(5)}$	6.38
0.3	$0.7378^{(9)}$	$0.6071^{(4)}$	$0.2184^{(5)}$	$0.6672^{(5)}$	$0.9827^{(6)}$	$0.9881^{(3)}$	$0.9689^{(2)}$	$0.9856^{(4)}$	4.75
0.4	$0.7752^{(6)}$	$0.5786^{(7)}$	$0.2149^{(7)}$	$0.6690^{(3)}$	$0.9846^{(3)}$	$0.9872^{(4)}$	$0.9651^{(4)}$	$\mathbf{0.9862^{(1)}}$	4.38
0.5	$0.7637^{(7)}$	$0.6009^{(5)}$	$0.2217^{(3)}$	$0.6758^{(2)}$	$0.9838^{(5)}$	$0.9841^{(6)}$	$0.9568^{(6)}$	$0.9840^{(6)}$	5.00
0.6	$0.7810^{(4)}$	$0.5621^{(8)}$	$0.2098^{(8)}$	$0.6627^{(8)}$	$0.9802^{(10)}$	$0.9733^{(8)}$	$0.9294^{(8)}$	$0.9760^{(9)}$	7.88
0.7	$0.7781^{(5)}$	$0.5938^{(6)}$	$0.2219^{(2)}$	$0.6686^{(4)}$	$0.9879^{(2)}$	$0.9712^{(9)}$	$0.9248^{(9)}$	$0.9677^{(11)}$	6.00
0.8	$0.8012^{(2)}$	$0.5421^{(9)}$	$0.2066^{(9)}$	$0.6612^{(9)}$	$0.9821^{(7)}$	$0.9653^{(11)}$	$0.9102^{(11)}$	$0.9718^{(10)}$	8.50
0.9	$0.7954^{(3)}$	$0.5316^{(10)}$	$0.2018^{(10)}$	$0.6529^{(11)}$	$\mathbf{0.9885^{(1)}}$	$0.9681^{(10)}$	$0.9175^{(10)}$	$0.9761^{(8)}$	7.88
1	$\mathbf{0.8271^{(1)}}$	$0.5095^{(11)}$	$0.2006^{(11)}$	$0.6556^{(10)}$	$0.9806^{(9)}$	$0.9765^{(7)}$	$0.9374^{(7)}$	$0.9781^{(9)}$	7.88

Appendix 6: The CCS performance of the DQN-BSPER model under 11 kinds of k_1/k_2

	IF dataset				LE dataset				Average rank
k_1/k_2	*TPR*	*ACC*	*F1*	*AUC*	*TPR*	*ACC*	*F1*	*AUC*	
6/1	$\mathbf{0.9422^{(1)}}$	$0.1829^{(11)}$	$0.1461^{(7)}$	$0.5321^{(6.5)}$	$0.9885^{(2)}$	$0.9636^{(11)}$	$0.9069^{(10)}$	$0.9734^{(11)}$	7.44
5/1	$0.9412^{(2)}$	$0.2064^{(10)}$	$0.1506^{(6)}$	$0.5321^{(6.5)}$	$0.9803^{(6)}$	$0.9696^{(10)}$	$0.9063^{(11)}$	$0.9769^{(6)}$	7.19
4/1	$0.8947^{(3)}$	$0.2362^{(9)}$	$0.1581^{(3)}$	$0.5519^{(5)}$	$0.9825^{(5)}$	$0.9739^{(8)}$	$0.9293^{(9)}$	$0.9812^{(4)}$	5.75

k_1/k_2	IF dataset				LE dataset				Average rank
	TPR	*ACC*	*F1*	*AUC*	*TPR*	*ACC*	*F1*	*AUC*	
3/1	0.8674$^{(4)}$	0.3998$^{(8)}$	0.1557$^{(4)}$	0.5608$^{(4)}$	**0.9896$^{(1)}$**	0.9780$^{(7)}$	0.9416$^{(7)}$	0.9825$^{(3)}$	4.75
2/1	0.8096$^{(5)}$	0.5477$^{(7)}$	0.1547$^{(5)}$	0.5827$^{(2)}$	0.9841$^{(3)}$	0.9829$^{(6)}$	0.9644$^{(5)}$	0.9828$^{(2)}$	4.38
1	0.7464$^{(6)}$	0.6296$^{(6)}$	**0.2260$^{(1)}$**	**0.6779$^{(1)}$**	0.9838$^{(4)}$	0.9888$^{(4)}$	**0.9690$^{(1)}$**	**0.9857$^{(1)}$**	**3.00**
1/2	0.1830$^{(7)}$	0.8276$^{(5)}$	0.1928$^{(2)}$	0.5694$^{(3)}$	0.9702$^{(8)}$	0.9883$^{(5)}$	0.9617$^{(6)}$	0.9783$^{(5)}$	5.13
1/3	0.0208$^{(11)}$	0.9271$^{(4)}$	0.0395$^{(11)}$	0.5092$^{(10)}$	0.9786$^{(7)}$	0.9730$^{(9)}$	0.9299$^{(8)}$	0.9752$^{(8)}$	8.50
1/4	0.0519$^{(8)}$	0.9294$^{(3)}$	0.0874$^{(8)}$	0.5205$^{(8)}$	0.9601$^{(9)}$	0.9908$^{(3)}$	0.9678$^{(2)}$	0.9747$^{(9)}$	6.25
1/5	0.0403$^{(9)}$	**0.9315$^{(1.5)}$**	0.0711$^{(9)}$	0.5163$^{(9)}$	0.9554$^{(10)}$	0.9911$^{(2)}$	0.9653$^{(3)}$	0.9734$^{(10)}$	6.69
1/6	0.0317$^{(10)}$	**0.9315$^{(1.5)}$**	0.0576$^{(10)}$	0.5079$^{(11)}$	0.9544$^{(11)}$	**0.9913$^{(1)}$**	0.9649$^{(4)}$	0.9768$^{(7)}$	6.94

Appendix 7: The customer credit scoring performance of DQN-BSPER and seven traditional benchmark classification models on the IF and LE datasets

		DQN-BSPER	DNN	LDA	LR	NB	DT	KNN	SVM
IF	*TPR*	**0.7464$^{(1)}$**	0.1826$^{(4)}$	0.3129$^{(2)}$	0.0367$^{(8)}$	0.0754$^{(6)}$	0.1113$^{(5)}$	0.2084$^{(3)}$	0.0742$^{(7)}$
	ACC	0.6296$^{(5)}$	0.6213$^{(6)}$	0.9229$^{(3)}$	0.9256$^{(2)}$	0.1012$^{(7)}$	0.8717$^{(4)}$	0.9287$^{(1)}$	0.0752$^{(8)}$
	F1	**0.2260$^{(1)}$**	0.0237$^{(7)}$	0.0555$^{(6)}$	0.0214$^{(8)}$	0.1401$^{(2)}$	0.1070$^{(4)}$	0.0567$^{(5)}$	0.1381$^{(3)}$
	AUC	**0.6779$^{(1)}$**	0.5543$^{(4)}$	0.6201$^{(2)}$	0.5539$^{(5)}$	0.5171$^{(7)}$	0.5198$^{(6)}$	0.5678$^{(3)}$	0.5013$^{(8)}$
IF (ROS)	*TPR*	**0.6672$^{(1)}$**	0.0822$^{(6)}$	0.1009$^{(5)}$	0.1011$^{(4)}$	0.0753$^{(8)}$	0.1489$^{(2)}$	0.1095$^{(3)}$	0.0757$^{(7)}$
	ACC	0.6343$^{(3)}$	0.2365$^{(6)}$	0.5921$^{(4)}$	0.5883$^{(5)}$	0.0985$^{(8)}$	**0.8446$^{(1)}$**	0.7719$^{(2)}$	0.1200$^{(7)}$
	F1	**0.2127$^{(1)}$**	0.1507$^{(5)}$	0.1890$^{(2)}$	0.1597$^{(4)}$	0.1399$^{(8)}$	0.1806$^{(3)}$	0.1435$^{(6)}$	0.1403$^{(7)}$
	AUC	**0.6514$^{(1)}$**	0.5221$^{(6)}$	0.5320$^{(3)}$	0.5224$^{(5)}$	0.5165$^{(7)}$	0.5421$^{(2)}$	0.5257$^{(4)}$	0.5108$^{(8)}$
IF (RUS)	*TPR*	**0.7418$^{(1)}$**	0.0900$^{(7)}$	0.1106$^{(3)}$	0.1096$^{(4)}$	0.0994$^{(5)}$	0.0981$^{(6)}$	0.1110$^{(2)}$	0.0750$^{(8)}$
	ACC	**0.6243$^{(1)}$**	0.5270$^{(6)}$	0.5883$^{(4)}$	0.5917$^{(3)}$	0.4795$^{(7)}$	0.5699$^{(5)}$	0.5919$^{(2)}$	0.0985$^{(8)}$
	F1	**0.2199$^{(1)}$**	0.1561$^{(6)}$	0.1958$^{(2)}$	0.1867$^{(3)}$	0.1615$^{(5)}$	0.1679$^{(4)}$	0.1435$^{(7)}$	0.1394$^{(8)}$
	AUC	**0.6645$^{(1)}$**	0.5152$^{(7)}$	0.5357$^{(3)}$	0.5308$^{(4)}$	0.5259$^{(5)}$	0.5212$^{(6)}$	0.5368$^{(2)}$	0.5139$^{(8)}$
LE	*TPR*	**0.9838$^{(1)}$**	0.9824$^{(2)}$	0.9814$^{(4)}$	0.9823$^{(3)}$	0.9745$^{(5)}$	0.9642$^{(6)}$	0.9487$^{(7)}$	0.1943$^{(8)}$
	ACC	**0.9888$^{(1)}$**	0.9881$^{(2)}$	0.9475$^{(5)}$	0.9833$^{(4)}$	0.9343$^{(6)}$	0.9874$^{(3)}$	0.9277$^{(7)}$	0.2569$^{(8)}$
	F1	**0.9690$^{(1)}$**	0.9658$^{(2)}$	0.8287$^{(5)}$	0.9513$^{(4)}$	0.7803$^{(6)}$	0.9650$^{(3)}$	0.7578$^{(7)}$	0.3253$^{(8)}$
	AUC	**0.9857$^{(1)}$**	0.9810$^{(3)}$	0.9692$^{(5)}$	0.9815$^{(2)}$	0.9517$^{(6)}$	0.9783$^{(4)}$	0.9368$^{(7)}$	0.5966$^{(8)}$
LE (ROS)	*TPR*	**0.9827$^{(1)}$**	0.2322$^{(7)}$	0.9783$^{(3)}$	0.9825$^{(2)}$	0.9691$^{(4)}$	0.9648$^{(5)}$	0.6091$^{(6)}$	0.1972$^{(8)}$
	ACC	**0.9883$^{(1)}$**	0.4075$^{(7)}$	0.9765$^{(4)}$	0.9881$^{(2)}$	0.9339$^{(5)}$	0.9870$^{(3)}$	0.8721$^{(6)}$	0.2703$^{(8)}$
	F1	**0.9674$^{(1)}$**	0.3768$^{(7)}$	0.9313$^{(4)}$	0.9672$^{(3)}$	0.7794$^{(5)}$	0.9674$^{(2)}$	0.6913$^{(6)}$	0.3294$^{(8)}$
	AUC	**0.9860$^{(1)}$**	0.6158$^{(7)}$	0.9772$^{(4)}$	0.9852$^{(2)}$	0.9491$^{(5)}$	0.9784$^{(3)}$	0.7810$^{(6)}$	0.5981$^{(8)}$
LE (RUS)	*TPR*	0.9790$^{(2)}$	0.3266$^{(7)}$	0.9771$^{(3)}$	**0.9938$^{(1)}$**	0.9720$^{(4)}$	0.8969$^{(5)}$	0.6278$^{(6)}$	0.1932$^{(8)}$
	ACC	**0.9844$^{(1)}$**	0.6321$^{(7)}$	0.9763$^{(4)}$	0.9832$^{(3)}$	0.9340$^{(5)}$	0.9844$^{(2)}$	0.8813$^{(6)}$	0.2516$^{(8)}$
	F1	0.9575$^{(2)}$	0.4912$^{(7)}$	0.9307$^{(4)}$	**0.9632$^{(1)}$**	0.7796$^{(5)}$	0.9342$^{(3)}$	0.6913$^{(6)}$	0.3238$^{(8)}$
	AUC	**0.9823$^{(1)}$**	0.6615$^{(7)}$	0.9766$^{(3)}$	0.9818$^{(2)}$	0.9504$^{(4)}$	0.9456$^{(5)}$	0.7939$^{(6)}$	0.5962$^{(8)}$

	DQN-BSPER	DNN	LDA	LR	NB	DT	KNN	SVM
Average rank	**1.33**	5.63	3.63	3.50	5.63	3.83	4.83	7.63

Acknowledgements The financial support from the National Natural Science Foundation of China (Grant No. 72171160), the Guangdong Province Philosophy and Social Science Planning Project (Grant No. GD23YGL24), the Excellent Youth Foundation of Sichuan Province (Grant No. 2020JDJQ0021), the Tianfu Ten-Thousand Talents Program of Sichuan Province (Grant No. 0082204151153), the Humanities and Social Science Youth Foundation of Ministry of Education of China (Grant No. 23YJCZH088), the Sichuan Science and Technology Program (Grant No. 2023YFQ0018), the Scientific Research Starting Project of SWPU (Grant No. 2021QHZ020) are gratefully acknowledged.

Author contributions Yadong Wang: wrote the original draft, reviewed, formal analyzed, investigated, visualizated, validated. Yanlin Jia: Investigated, supervized. Yuhang Tian: reviewed, edited, validated. Jin Xiao: reviewed, formal analyzed, Investigated, Fund.

Declarations

Conflict of interest The authors declare no competing interests.

References

Altman EI (1968) Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. J Finance 23(4):589–609

Baesens B, Van Gestel T, Viaene S, Stepanova M, Suykens J, Vanthienen J (2003) Benchmarking state-of-the-art classification algorithms for credit scoring. J Oper Res Soc 54(6):627–635

Bastani K, Asgari E, Namavari H (2019) Wide and deep learning for peer-to-peer lending. Expert Syst Appl 134:209–224

Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 6(1):20–29

Blumenstock G, Lessmann S, Seow H-V (2022) Deep learning for survival and competing risk modelling. J Oper Res Soc 73(1):26–38

Borgonovo E, Smith CL (2011) A study of interactions in the risk assessment of complex engineering systems: an application to space PSA. Oper Res 59(6):1461–1476

Bradley AP (1997) The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognit 30(7):1145–1159

Cai R, Li H, Wang S, Chen C, Kot A (2020) DRL-FAS: a novel framework based on deep reinforcement learning for face anti-spoofing. IEEE Trans Inf Forensics Secur 16:937–951

Chatterjee M, Namin A-S (2019) Detecting phishing websites through deep reinforcement learning. In: Proceedings of the IEEE 43rd annual computer software and applications conference, 2019. IEEE, pp 227–232

Chen S-Y, Yu Y, Da Q, Tan J, Huang H-K, Tang H-H (2018) Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, 2018. ACM, pp 1187–1196

Crone SF, Finlay S (2012) Instance sampling in credit scoring: an empirical study of sample size and balancing. Int J Forecast 28(1):224–238
Dastile X, Celik T, Potsane M (2020) Statistical and machine learning models in credit scoring: a systematic literature survey. Appl Soft Comput 91:106263
De Moor BJ, Gijsbrechts J, Boute RN (2022) Reward shaping to improve the performance of deep reinforcement learning in perishable inventory management. Eur J Oper Res 301(2):535–545
Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(Jan):1–30
Ding Y, Ma L, Ma J, Suo M, Tao L, Cheng Y, Lu C (2019) Intelligent fault diagnosis for rotating machinery using deep Q-network based health state classification: a deep reinforcement learning approach. Adv Eng Inform 42:100977
Du N, Li L, Lu T, Lu X (2020) Prosocial compliance in P2P lending: a natural field experiment. Manag Sci 66(1):315–333
Dumitrescu E, Hue S, Hurlin C, Tokpavi S (2022) Machine learning for credit scoring: improving logistic regression with non-linear decision-tree effects. Eur J Oper Res 297(3):1178–1192
Fan C, Zeng L, Sun Y, Liu Y-Y (2020) Finding key players in complex networks through deep reinforcement learning. Nat Mach Intell 2(6):317–324
Fernandes GB, Artes R (2016) Spatial dependence in credit risk and its improvement in credit scoring. Eur J Oper Res 249(2):517–524
Friedman M (1940) A comparison of alternative tests of significance for the problem of m rankings. Ann Math Stat 11(1):86–92
Gosavi A (2009) Reinforcement learning: a tutorial survey and recent advances. INFORMS J Comput 21(2):178–192
Gunnarsson BR, Vanden Broucke S, Baesens B, Óskarsdóttir M, Lemahieu W (2021) Deep learning for credit scoring: do or don't? Eur J Oper Res 295(1):292–305
Guo Y, Zhou W, Luo C, Liu C, Xiong H (2016) Instance-based credit risk assessment for investment decisions in P2P lending. Eur J Oper Res 249(2):417–426
Hosmer DW Jr, Lemeshow S, Sturdivant RX (2013) Applied logistic regression, vol 398. Wiley, Hoboken
Iman RL, Davenport JM (1980) Approximations of the critical region of the Fbietkan statistic. Commun Stat Theory Methods 9(6):571–595
Kou G, Peng Y, Wang G (2014) Evaluation of clustering algorithms for financial risk analysis using MCDM methods. Inf Sci 275:1–12
Kou G, Olgu Akdeniz Ö, Dinçer H, Yüksel S (2021a) Fintech investments in European banks: a hybrid IT2 fuzzy multidimensional decision-making approach. Financ Innov 7(1):39
Kou G, Xu Y, Peng Y, Shen F, Chen Y, Chang K, Kou S (2021b) Bankruptcy prediction for SMEs using transactional data and two-stage multiobjective feature selection. Decis Support Syst 140:113429
Lei K, Zhang B, Li Y, Yang M, Shen Y (2020) Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. Expert Syst Appl 140:112872
Lessmann S, Baesens B, Seow H-V, Thomas LC (2015) Benchmarking state-of-the-art classification algorithms for credit scoring: an update of research. Eur J Oper Res 247(1):124–136
Li H, Xu H (2020) Deep reinforcement learning for robust emotional classification in facial expression recognition. Knowl Based Syst 204:106172
Li Y, Wang X, Djehiche B, Hu X (2020) Credit scoring by incorporating dynamic networked information. Eur J Oper Res 286(3):1103–1112
Lim M, Abdullah A, Jhanjhi N (2021) Performance optimization of criminal network hidden link prediction model with deep reinforcement learning. J King Saud Univ Comput Inf Sci 33(10):1202–1210
Lin E, Chen Q, Qi X (2020) Deep reinforcement learning for imbalanced classification. Appl Intell 5:1–15
Liu Y, Chen Y, Jiang T (2020) Dynamic selective maintenance optimization for multi-state systems over a finite horizon: a deep reinforcement learning approach. Eur J Oper Res 283(1):166–181
Lopez-Martin M, Carro B, Sanchez-Esguevillas A (2020) Application of deep reinforcement learning to intrusion detection for supervised problems. Expert Syst Appl 141:112963
Loyola-González O, Martínez-Trinidad JF, Carrasco-Ochoa JA, García-Borroto M (2016) Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases. Neurocomputing 175:935–947
Luo B, Yang Y, Liu D (2018) Adaptive Q-Learning for data-based optimal output regulation with experience replay. IEEE Trans Cybern 48(12):3337–3348
Marqués AI, García V, Sánchez JS (2013) On the suitability of resampling techniques for the class imbalance problem in credit scoring. J Oper Res Soc 64(7):1060–1070
Martinez C, Ramasso E, Perrin G, Rombaut M (2020) Adaptive early classification of temporal sequences using deep reinforcement learning. Knowl Based Syst 190:105290

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing Atari with deep reinforcement learning. ArXiv preprint arXiv:1312.5602
Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG et al (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533
Óskarsdóttir M, Bravo C, Sarraute C, Vanthienen J, Baesens B (2019) The value of big data for credit scoring: enhancing financial inclusion using mobile phone data and social network analytics. Appl Soft Comput 74:26–39
Patel D, Hazan H, Saunders DJ, Siegelmann HT, Kozma R (2019) Improved robustness of reinforcement learning policies upon conversion to spiking neuronal network platforms applied to Atari Breakout game. Neural Netw 120:108–115
Petrides G, Moldovan D, Coenen L, Guns T, Verbeke W (2020) Cost-sensitive learning for profit-driven credit scoring. J Oper Res Soc 73(2):1–13
Protopapadakis E, Niklis D, Doumpos M, Doulamis A, Zopounidis C (2019) Sample selection algorithms for credit risk modelling through data mining techniques. Int J Data Min Model Manag 11(2):103–128
Rish I (2001) An empirical study of the naive Bayes classifier. Workshop Empir Methods Artif Intell 3(22):41–46
Schaul T, Quan J, Antonoglou I, Silver D (2015) Prioritized experience replay. ArXiv Preprint arXiv 1511:05952
Schnaubelt M (2022) Deep reinforcement learning for the optimal placement of cryptocurrency limit orders. Eur J Oper Res 296(3):993–1006
Serrano-Cinca C, Gutiérrez-Nieto B (2016) The use of profit scoring as an alternative to credit scoring systems in peer-to-peer (P2P) lending. Decis Support Syst 89:113–122
Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A et al (2018) A general reinforcement learning algorithm that masters chess, Shogi, and Go through self-play. Science 362(6419):1140–1144
So MM, Thomas LC (2011) Modelling the profitability of credit cards by Markov decision processes. Eur J Oper Res 212(1):123–130
Sun AY (2020) Optimal carbon storage reservoir management through deep reinforcement learning. Appl Energy 278:115660
Sutton R, Barto A (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
Tang Y, Zhang Y-Q, Chawla NV, Krasser S (2008) SVMs modeling for highly imbalanced classification. IEEE Trans Syst Man Cybern B 39(1):281–288
Trafalis TB, Gilbert RC (2006) Robust classification and regression using support vector machines. Eur J Oper Res 173(3):893–909
van Heeswijk W (2022) Strategic bidding in freight transport using deep reinforcement learning. Ann Oper Res. https://doi.org/10.1007/s10479-022-04572-z
Veganzones D, Séverin E (2018) An investigation of bankruptcy prediction in imbalanced datasets. Decis Support Syst 112:111–124
Wang H, Kou G, Peng Y (2021) Multi-class misclassification cost matrix for credit ratings in peer-to-peer lending. J Oper Res Soc 72(4):923–934
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 Syst Appl 200:117013
Watkins CJ, Dayan P (1992) Q-learning. Mach Learn 8(3–4):279–292
Wauters M, Vanhoucke M (2017) A nearest neighbour extension to project duration forecasting with artificial intelligence. Eur J Oper Res 259(3):1097–1111
Wilcoxon F (1992) Individual comparisons by ranking methods. In: Breakthroughs in statistics. Springer, Berlin, pp 196–202
Wurman PR, Barrett S, Kawamoto K, MacGlashan J, Subramanian K, Walsh TJ et al (2022) Outracing champion Gran Turismo drivers with deep reinforcement learning. Nature 602(7896):223–228
Xia Y, Zhao J, He L, Li Y, Niu M (2020) A novel tree-based dynamic heterogeneous ensemble method for credit scoring. Expert Syst Appl 159:113615
Xiao J, Zhou X, Zhong Y, Xie L, Gu X, Liu D (2020) Cost-sensitive semi-supervised selective ensemble model for customer credit scoring. Knowl Based Syst 189:105118
Xiao J, Wang Y, Chen J, Xie L, Huang J (2021) Impact of resampling methods and classification models on the imbalanced credit scoring problems. Inf Sci 569:508–526
Yeo B, Grant D (2018) Predicting service industry performance using decision tree analysis. Int J Inf Manag 38(1):288–300
Zhang G, Hu W, Cao D, Liu W, Huang R, Huang Q et al (2021) Data-driven optimal energy management for a wind–solar–diesel-battery-reverse osmosis hybrid energy system using a deep reinforcement learning approach. Energy Convers Manag 227:113608

Zhao D, Chen Y, Lv L (2016) Deep reinforcement learning with visual attention for vehicle classification. IEEE Trans Cogn Dev Syst 9(4):356–367