Migration Letters

Volume: 21, No: 3, pp. 1387-1404

ISSN: 1741-8984 (Print) ISSN: 1741-8992 (Online)
www.migrationletters.com

Algorithmic Loan Risk Prediction Method Based on PSO-EBGWO-Catboost

Suihai Chen[1], Bong Chih How[2], Po Chan Chiu[3]

Abstract

Loan risk analysis is a common challenge faced by global financial institutions. Under the background of big data, it is of practical significance to prevent loan risk by the machine learning algorithm. Aiming at the characteristics of unbalanced loan data and high noise, this paper proposes an improved Gray Wolf optimization strategy (PSO-EBGWO). PSO-EBGWO is used to optimize the parameters of the CatBoost model. In this method, the Gray Wolf optimized algorithm (EBGWO) is further optimized by particle swarm optimization (PSO), and when combined with it, the convergence performance of the model is improved, the parameters of the model are reduced, and the model is simplified. To a certain extent, it avoids the inefficiency of the Gray Wolf algorithm, balances the ability of local search and global development, and improves the accuracy of the model. Compared with the traditional credit evaluation model, PSO-EBGWO-CatBoost has better accuracy and practical application value.

Keywords: *loan risk; EBGWO; PSO; CatBoost; Accuracy.*

1. INTRODUCTION

With the rapid development of financial economy in our country, credit business has become the main business of banks and financial companies. The quality of customer credit directly affects the business performance of financial companies. To improve their ability to compete in the market, financial companies need to be able to distinguish customers who are likely to be a good investment and those who are not. By identifying and lending to the right customers, they can increase their profits. In fact, there is a large group of customers who fall between being good or bad, and traditional risk assessment methods cannot tell them apart effectively. This is because, the logistic regression algorithm used in traditional risk control model has low accuracy and ability in distinguish good customers from bad customers even though it possesses high speed, high stability, strong explanatory power.

Enhancing the accuracy of a model with high speed, stability, and clear interpretability is the key focus for improving traditional risk control algorithms. The logistic regression algorithm used in the traditional risk control model has the advantages of fast training speed, easy understanding and good model interpretability (Stoltzfus, 2011)[1]. It is suitable for solving linear problems, but its shortcomings are also obvious, such as low accuracy and inability to deal with nonlinear data. Kaastra and Boyd (Kaastra, Boyd, 1996)[2] put forward that the ideal risk control model is able to quickly distinguish good customers (good credit) and bad customers (bad credit) on the premise of good accuracy,

[1] Computer Science and Information Technology, Universiti Malaysia Sarawak (UNIMAS), Sarawak, Borneo, Malaysia
[2] Computer Science and Information Technology, Universiti Malaysia Sarawak (UNIMAS), Sarawak, Borneo, Malaysia
[3] Computer Science and Information Technology, Universiti Malaysia Sarawak (UNIMAS), Sarawak, Borneo, Malaysia

strong stability, strong interpretability, good universality and low consumption of computing resources (enterprise cost). The frequent opinion found is, machine learning algorithms are better than logistic regression algorithms. Many scholars (Ma, Wang, Yang, 2018)[3] currently use XGBoost and LightGBM algorithms for data testing, and the results are significantly better than the traditional logistic regression algorithm. Many scholars (Chang, Chang, 2018)[4] suggest using XGBoost instead of logistic regression. A few scholars (Ma, Sha, Wang, 2018)[5] have also combined logistic regression algorithm with XGBoost algorithm and achieved some results. However, XGBoost also has its drawbacks, such as overfitting, difficulty in parameter tuning, slow learning speed, and LightGBM's algorithm is not as accurate as XGBoost's [6](Liang, Luo, Zhao, 2020). The new CatBoost algorithm compensates for the shortcomings of XGBoost and LightGBM in many ways.

CatBoost algorithm has high accuracy, interpretability and time complexity. Therefore, many experts begin to study it to replace the traditional risk control model algorithm. However, the CatBoost algorithm also has its shortcomings. The algorithm needs to adjust its abundant parameters in order to have high accuracy. Since it has many parameters, so it is impossible to manually adjust one by one. Therefore, it is necessary to use other algorithms to help CatBoost algorithm to automatically Determine the ideal parameters for the Grey Wolf Optimizer (GWO) algorithm introduced by Seyedali Mirjalili. (2014)[7] 2014 has a simple structure and strong advantages in finding the optimal parameters. How to combine these two algorithms to make GWO algorithm quickly find the optimal parameters of CatBoost algorithm, while avoiding GWO algorithm falling into local optimal solution, and balancing the exploration and development ability of the algorithm needs further research.

How to avoid the random search behavior of GWO in the parameter search process, avoid the algorithm falling from local optimum, realize the balance between exploration and exploitation ability, and find the most suitable parameters for the CatBoost algorithm.The main processes for GWO optimization are as follows: The first step requires the improvement of the GWO algorithm, referring to as the "EBGWO" algorithm. The EBGWO algorithm is an enhanced version of the Grey Wolf Optimization algorithm, possessing strong global search capabilities and, to some extent, mitigating the local convergence issues associated with the Grey Wolf Optimization algorithm. It can effectively enhance the predictive accuracy and execution speed of the CatBoost algorithm model. The second step is that although the optimized GWO algorithm (EBGWO) can effectively avoid premature convergence and local optimization, its lack does not consider individual experience and lacks communication between individual and group positions [8] . The core principle of the Particle Swarm Optimization (PSO) algorithm involves the continuous movement of particles at variable speeds within a space. These particles navigate by leveraging their individual memories and group communication to determine the next position. Each particle adjusts its trajectory in pursuit of the optimal individual position. (Reddy, Viswanath, 2018)[9].

Hence, by incorporating the update mechanism of particle positions, the Gray Wolf algorithm can imbue a level of memory in its optimization process, effectively replacing the individual position update for Gray Wolves, and the EBGWO algorithm has the communication ability between the individual and the group position, so as to solve the problem. The following chapters will introduce the detailed process of PSO-EBGWO-CatBoost algorithm, and carry out experimental test verification.

2、Particle Swarm Optimization Hybrid Grey Wolf Optimization (PSO-EBGWO)

2.1、Grey Wolf Optimization(EBGWO)

2.1.1 GWO algorithm principle

Inspired by the searching and predation behavior of gray wolves, variants in GWO mimic the hunting and leadership mechanisms of wolves[10]. The variants are divided into four different groups according to the rank of wolves in nature, namely α, β, δ and γ. Gray wolves like to live in packs, and their leader is usually a male gray Wolf called alpha, who decides the rest and hunting of the pack and other social behaviors. It is regarded as the Wolf closest to the optimal solution, which is the highest rank in the pack[11]. The next highest level is called the beta Wolf, and it helps the alpha Wolf to make decisions or help reinforce the alpha Wolf's command[12]. The βwolves should respect the αwolves, but command the rest of the lower rank pack and give the information back to the αwolves. The lowest ranking gray Wolf is γ, which acts as a scapegoat Wolf and always obeys other higher-ranking wolves to help make up the hierarchy of the pack. Individuals that do not belong to any of the three species are called delta wolves, and they take orders from alpha and beta wolves but dominate gamma wolves[13]. In addition to mimicking the gray Wolf hierarchy, GWO also introduces Wolf pack behaviors such as Engaging in the optimization process involves the exploration for prey, encircling identified targets, and executing attacks on the prey.

The group follows α,β,δto search for prey, and is used to represent the forced separation of gray wolves from prey to determine the optimal attack target. After determining the target of the attack, namely, the surrounding behavior of wolves can be expressed as follows:

$$\begin{aligned} \vec{D} &= |C * \vec{X}_{p(t)} - \vec{X}(t)| \\ \vec{X}(t+1) &= \vec{X}_{p(t)} - \vec{A} * \vec{D} \\ \vec{A} &= 2\vec{\alpha} * \vec{r}_1 - \vec{\alpha} \\ \vec{C} &= 2 * \vec{r}_2 \end{aligned} \tag{2-1}$$

Where is the Euclidean distance between $\vec{D}$ gray Wolf and prey; $\vec{X}(t)$ is the position vector after the movement of the gray Wolf. The position vector after $\vec{X}_{p(t)}$ times; In the bounding process, the coefficient $\vec{\alpha}$ decreases linearly from 2 to 0. The values of $\vec{r}_1$ and $\vec{r}_2$ modulus vary randomly between [0,1].

After encircling prey α,β,δare considered as three potential solutions, and their positions change with the movement of prey. The chasing behavior of wolves can be expressed as follows:

$$\begin{aligned} \vec{D}_j &= |\vec{C}_n * \vec{X}_j - \vec{X}| \\ \vec{X}_n &= \vec{X}_j - \vec{A}_n * \vec{D}_j \\ \vec{X}_p(t+1) &= \frac{\sum_n \vec{X}_n}{3} \end{aligned} \tag{2-2}$$

Where j =α, β, δ; N = 1,2,3; $\vec{D}$ represents the Euclidean distance from α, β,δ to γ; $\vec{X}$i defines the step size and direction of γ approaching α, β and δ. $\vec{X}$i(t+1) is the final position of γ.

When the prey stops moving, the gray Wolf attacks the prey, which determines the optimal value. The decline of the value of $\vec{\alpha}$ from 2 to 0 is the core of this stage, indicating that the value of $\vec{A}$ changes with in the corresponding interval, so the next

update position of gray Wolf will be closer to the prey position (the optimal solution).

2.1.2 EBGWO algorithm principle and pseudo-code

The number of leader wolves in the original GWO algorithm is fixed to three, namely α, β, δ. As can be seen from formula (2-7) in Section 2.1.1, X (t+1) represents the updated position of the Wolf, which is calculated by X(t+1)=(X1+X2+X3)/3, and X is the position vector of the gray Wolf. If the amount of data is very large (the amount of computation is very large), the data samples are irregular, the data segments are somewhat similar and high, and some are scattered and irregular. When the attributes of data samples are easy to distinguish good customers from bad customers, only two leading wolves are needed for calculation (X1 and X2), which can improve the convergence speed of the algorithm and reduce the number of iterations. When the data is between good and bad customer samples, the search scope needs to be expanded, and the number of leader wolves needs to be increased, that is, the calculation of X1, X2 and X3 is not enough, and an X4 needs to be added. However, the original GWO algorithm performs the calculation of three leading wolves regardless of the data situation, which makes the algorithm inefficient and needs to be further optimized.

According to the problem mentioned above, when the number of leaders increases to four, that is, α guides β, δ and γ to attack the prey, where α, β, δ and γ represent the currently obtained four optimal solutions, and the new individual update formula is as follows:

$$\begin{aligned} D_\alpha &= | C_1 \circ X_\alpha - X |, \\ D_\beta &= | C_2 \circ X_\beta - X |, \\ D_\sigma &= | C_2 \circ X_\sigma - X \\ D_\gamma &= | C_4 \circ X_\gamma - X |. \end{aligned} \qquad (2-3)$$

$$\begin{aligned} X_1 &= X_\alpha - A_1 \circ D_\alpha, \\ X_2 &= X_\beta - A_2 \circ D_\beta, \\ X_3 &= X_\delta - A_3 \circ D_\delta, \\ X_4 &= X_\gamma - A_4 \circ D_\gamma, \end{aligned} \qquad (2-4)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3 + X_4}{4} \qquad (2-5)$$

Among them, D_α, D_β, D_δ, D_γ, represent the distances between α wolf,β wolf, δwolf, γ wolf, and other members of the wolf pack, respectively. X is the current position vector of gray wolf; X_α、X_β、X_δ、X_γ are the positions of α wolf, β wolf, δ wolf and γ wolf respectively. A and C are synergy coefficient vectors; A decreases linearly from 2 to 0 in the whole iterative process; r1 and r2 are random vectors in [0,1]. X(t+1) is the updated position of the wolf.

Similarly, reducing the number of leader wolves means that α wolves guide β wolves to attack prey, and the new individual update formula is:

$$\begin{aligned} D_\alpha &= | C_1 \circ X_\alpha - X |, \\ D_\beta &= | C_2 \circ X_\beta - X |, \end{aligned} \qquad (2-6)$$

$$\begin{aligned} X_1 &= X_\alpha - A_1 \circ D_\alpha, \\ X_2 &= X_\beta - A_2 \circ D_\beta. \end{aligned} \qquad (2-7)$$

$$X(t+1) = \frac{X_1 + X_2}{2} \qquad (2-8)$$

As previously mentioned, when the number of leaders is 2 and 4, the corresponding formulas are (2-5) and (2-8). The formula for calculating the value of X(t+1) is not fixed but is automatically selected based on the actual situation for the number of leaders. Replace the original GWO algorithm formula X(t+1)=(X1+X2+X3)/3 with X(t+1)=(X1+X2)/2 or X(t+1)=(X1+X2+X3+X4)/4, X(t+1) is replaced by two or four is based on |A| value judgment. Wolves development (hunting), and search process is carried out according to the absolute value of |A|, As shown in figure 2.1, when |A|≥1 shows the current or A prey's position is not very good, you need to search its prey, It shows that the current calculation amount cannot obtain the optimal solution, that is to say, the three leadership classes of the original GWO algorithm cannot calculate the optimal solution and need to recalculate. It also shows that the differentiation between good and bad customers of the current data sample is very fuzzy, and further calculation is needed to increase, so the number of leader wolves needs to be increased. When the position of the population of prey |A| < 1 is better, need to reduce the number of leadership Wolf, improve the operation efficiency of the algorithm. According to the research of experts and scholars, the accuracy of GWO algorithm is worse when the leader Wolf is bigger or smaller (Heidari, bbaspour, et al. 2019)[14], the original GWO algorithm needs to appropriately reduce or increase the number of leader wolves to obtain the optimal result. Therefore, this study is based on the original GWO algorithm to increase or decrease the number of one leader Wolf, When |A|≥1, leadership wolf of the increase in the number 1, X(t+1) is 4 leadership Wolf; When the |A|<1, led the Wolf to reduce the number of 1, X(t+1) is 2 leadership Wolf, mathematics formula modified algorithm is as follows (2-9).

$$X(t+1)=\begin{cases} X(t+1)=\dfrac{X_1+X_2+X_3+X_4}{2} & if\ |A|\geq 1 \\ \dfrac{X_1+X_2}{2} & else \end{cases} \tag{2-9}$$

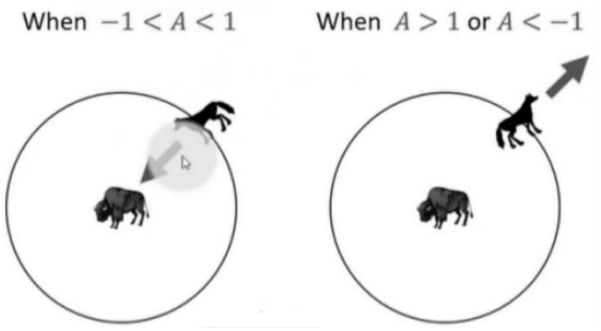

Figure 2.1: Conditions for attacking prey

The pseudo-code of the PSO-EBGWO algorithm is shown in Figure 2.2

Algorithm: EBGWO

1 Initialize the grey wolf population Xi (i=1,2,,,,n)

2 Initialize a,A,C,t=0

3 Calculate the fitness of each search agent

4 X_α=the best search agent

5 X_{β}=the second best search agent

6 X_{δ}= the third best search agent

7 X_{γ}= the fourth four best search agent

8 While (t< Max number of iterations)

9 For each seanch agent

10 Update the position of the current search agent

11end for

12 Update a,A,and C

13 Calculate the fitness of all search agents

14 If (|A|<1):

15 Use formula(2-6)update Xi

16 else:

17 Use formula(2-3)update Xi

18 t=t+1

19 end while

20 return X_{α}

Figure 4.1:EBGWO algorithm pseudo code

2.2、Particle Swarm optimization（PSO）

The fundamental concept of the Particle Swarm Optimization (PSO) algorithm lies in the continuous movement of particles within a space, adjusting their directions and speeds. These particles determine their next positions by leveraging both individual memory and group communication. [15], In pursuit of discovering the optimal solution, the formula for updating speed and position is employed.:

$$\begin{cases} v_i^{k+1}=v_i^k+c_1r_1\left(p_{best}^k-x_i^k\right)+c_2r_2\left(g_{best}-x_i^k\right) \\ x_i^{k+1}=x_i^k+v_i^{k+1} \end{cases} \quad (2\text{-}10)$$

2.3、Position updating formula based on “PSO-EBGWO”

The analysis above indicates that in the Particle Swarm Optimization (PSO) algorithm, individual particles collectively converge towards the optimal individual. While this results in fast convergence, it also highlights a limitation in terms of the algorithm's global balancing capability, This deficiency results in the algorithm's inability to discover the global optimal solution when tackling high-dimensional complex functions. Throughout the optimization process, Enhanced Grey Wolf Optimizer (EBGWO) neglects individual experiences and lacks effective communication between individual and group positions, which may lead to premature convergence of the algorithm and easy to fall into

local optimum. Most of the existing hybrid algorithms use sequential optimization objectives, which can not be well combined between algorithms[16]. Hence, PSO-EBGWO operates akin to a parallel algorithm, concurrently addressing both the convergence speed and accuracy considerations. The replacement of Gray Wolf individual position updates with particle position updates imparts a memory aspect to the Gray Wolf algorithm during the optimization search. This is achieved by fine-tuning the inertia constant ω, the coordinated hybrid algorithm balances the global search and local development capabilities, and the variation range of ω is [0.5,1]. Change equation (2-10) to equation (2-11):

$$\begin{aligned} v_i^{k+1} &= \omega * [v_i^k + c_1 r_1 (p_{best}^k - x_i^k) + c_2 r_2 (g_{best} - x_i^k) \\ &+ c_3 r_3 (x_3 - x_i^k)] \\ x_i^{k+1} &= x_i^k + v_i^{k+1} \end{aligned} \tag{2-11}$$

The first formula in equation (2-2) becomes:

$$\vec{D}_j = | C_n * \vec{X}_j - \omega \cdot \vec{X} | \tag{2-12}$$

2.4、PSO-EBGWO algorithm details

The PSO-EBGWO algorithm flow is shown in Figure.2.3

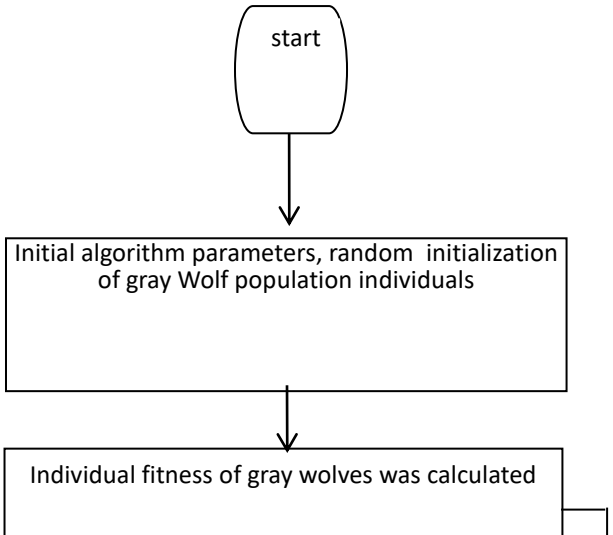

Fig.2.3 Flow chart of PSO-EBGWO algorithm

The stepwise execution of the hybridization of Particle Swarm Optimization with the Gray Wolf algorithm unfolds as follows:

Step 1:Initiate the parameters α, A, and C, and establish the gray Wolf population size N.;

Step 2: Randomly initialize individuals within the population and compute the fitness of each gray wolf.

Step 3: The three individuals with the highest adaptive values are denoted as α, β, and δ, with their corresponding positional information being Xα, Xβ, and Xδ, respectively.

Step 4: Adjust the values of A and C in accordance with Equation (2-11);

Step 5: Revise the individual positions of the gray wolves using Equation (2-12), and then proceed to Step 2 for recalculating and updating α, β, and δ once again;

Step 6: If the number of iterations (t) is equal to or exceeds the maximum allowable iterations (tmax), the gray wolves are deemed to have discovered the optimal solution position. Otherwise, it is regarded as a potential solution, and the algorithm returns to Step 3 to persist in the search for a more favorable position.

3. CATBOOST ALGORITHM AND PSO-EBGWO-CATBOOST IMPLEMENTATION PROCESS

3.1、 CatBoost algorithm

When it comes to feature engineering, the CatBoost algorithm uses a unique classification mode, integrates all features in a greedy strategy, calculates the statistical characteristics and frequency of the features, and generates specific derivative fields according to the hyperparameters set by itself. When the traditional gradient descent decision tree (GBDT) model trains the weak learner, the accuracy of the model is obtained based on the same dataset, which leads to the gradient estimation bias and finally leads to the prediction bias. The weak learning algorithm is as follows:

$$h_t = \underset{h \in H}{\operatorname{argmin}} E\left(-g^t(x,\mathrm{y}) - h(x)\right)^2 \tag{2-13}$$

Where is the weak learner of generation T and is the gradient of the loss function? Facing the common prediction offset the problem of the GBDT model, the CatBoost algorithm calculates the gradient of the loss function by proposing Ordered boosting, to obtain the unbiased gradient estimation, the algorithm trains a separate model through the training set that does not contain samples. For the model obtained by each sample, the algorithm uses the method of calculating the deviation to obtain the gradient estimate about the sample, which overcomes the prediction offset. For prediction, CatBoost uses the fully balanced tree as the basic predictor. Due to the symmetric structure of the fully balanced binary tree, its leaf node index can be encoded as a binary vector, the length is equal to the depth of the tree, and all features can be binarized to predict the model.

3.2、PSO-EBGWO-CatBoost implementation process

Therefore, PSO algorithm is used to optimize the parameters in CatBoost model, that is, the algorithm combination of PSO-EBGWO and CatBoost, that is, "PSO-EBGWO-Catboost" algorithm. The design process of this method is shown in the following figure.3.1

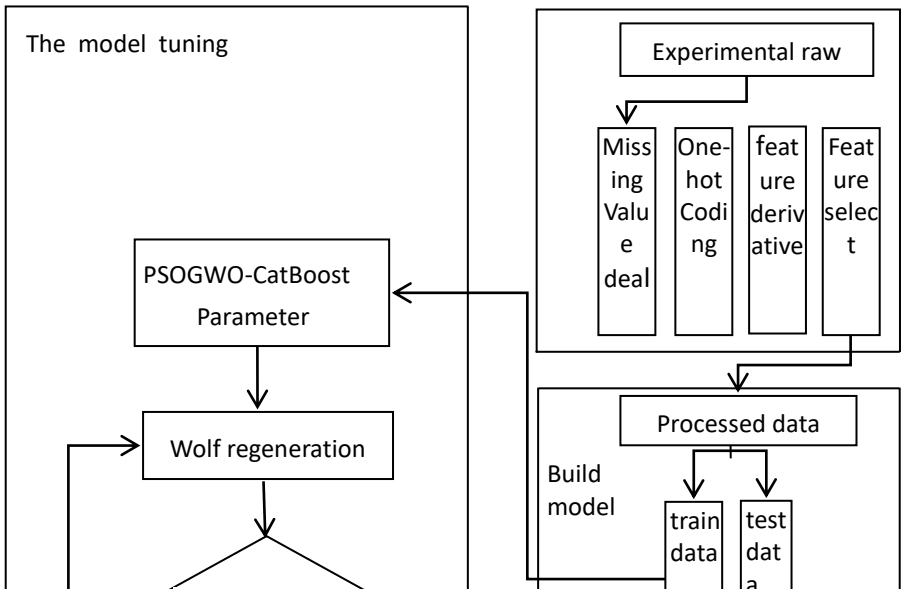

Fig.3.1 Flow chart of PSO-EBGWO algorithm

4、EXAMPLE AND ANALYSIS

4.1 Experimental of PSO-EBGWO algorithm

In order to more clearly explain the impact of the number of GWO and PSO-EBGWO algorithm leaders on algorithm performance, this part uses Shpere(Zhao and Yang, 2014)[17] and Schwefel(Yang, 2017)[18] The parameter Settings of the two functions are shown in Table 4.1. Since the original default number of leaders of the GWO algorithm is 3, and the PSO- EBGWO algorithm is mainly for the adjustment and optimization of the number of leaders, the experiment variables are mainly conducted according to the number of different leaders.

Table 4.1: Parameter Settings for Shpere and Schwefel functions

Function name	Expression	Rangeofvariable	D	N	T_{max}
Shpere	$F(\mathrm{x}) = \sum_{i=1}^{D} x_i^2$	[-100,100]	30	20	100
Schwefel	$F_8(\mathrm{x}) = \sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i \rvert})$	[-100,100]	30	20	100

4.2 Experimental results of Shpere and Schwefel functions

The Schwefel function, also known as the Schaefer function, is a common function optimization problem that can be used to measure the distance between each node in a given computer network. It is widely used to formalize and solve many complex optimization problems for comprehensive analysis and expression(Wang, Zhang etc, 2019)[19]. It can efficiently search for maximum or minimum values, making global optimal solutions easier to obtain. It helps computers to explicitly solve optimization problems, such as the least square method, and the algorithm can efficiently search for the global optimal solution. And more information can be gathered in one iteration. When the

algorithm is run on a specific data set, it can effectively identify the existing optimal solution. It can save the computing resources of the computer and reduce the computing time. Therefore, Schwefel function is an important optimization function, which can effectively help to solve optimization problems. It can improve the accuracy and efficiency of solving optimization problems, as well as the efficiency of some system design problems. As long as it is studied in the right direction, the Schwefel function can provide superior performance and make the system more efficient.

Shpere function, also known as spherical function, is a common minimization function used to optimize and minimize the results in multivariate functions, and the best results have been obtained. Its function is a function of space, describing points and objects in the coordinate plane, so it is suitable to find Determine the function's minimum and maximum values, as the GWO algorithm's mathematical formulation involves the computational process of vectors, which belongs to the calculation of dimensional vectors, Schwefel function and Shpere function are more suitable for solving the extreme value optimization problem of coordinate system and have great advantages (Chi, Su, etc, 2019)[20] Therefore, this section uses these two functions to test the performance of the GWO algorithm.

Since the number of iterations is unknown, 0-100 experimental tests are performed first, and then the number of iterations is selected according to the results. The test result using the Shpere function and Schwefel function is shown in Figure 4.1:

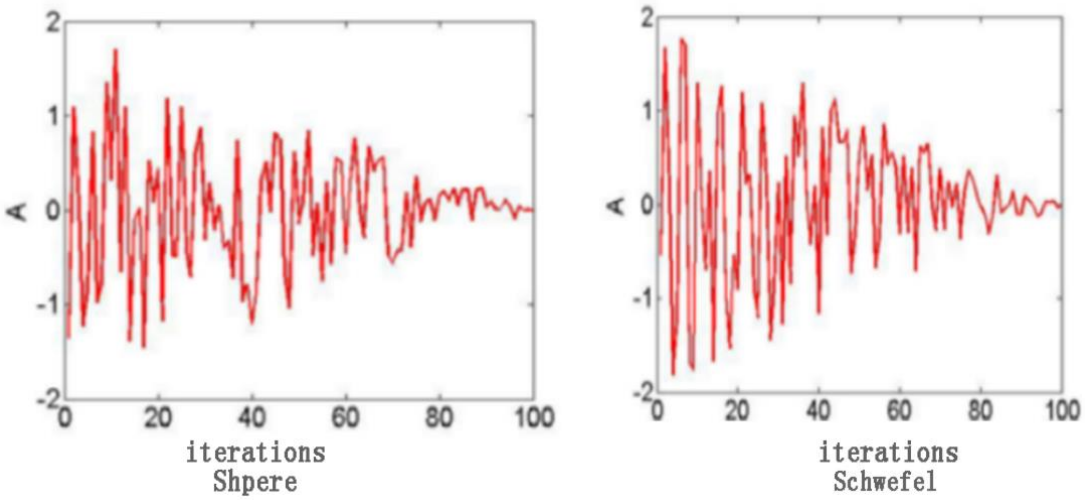

Figure 4.1: Test of Shpere and Schwefel Function A

It can be seen from the formula (2-7) in Section 2.1 that the value of A directly affects the update position of the leadership wolves, so the stability of the value of A means the stability of the leadership wolves. From the above figure, it can be known that setting the number of iterations to 100 is reasonable. Figure 4.1-4.3 shows the results of two test functions when the number of leaders is 3 (the number of leaders is 3 in the original GWO value). These include the search history of the gray Wolf in 100 iterations, the change of parameter A in 100 iterations, the trajectory of the first variable of the first Wolf in 100 iterations, the average fitness value of all wolves in 100 iterations, and the optimal value in each iteration.

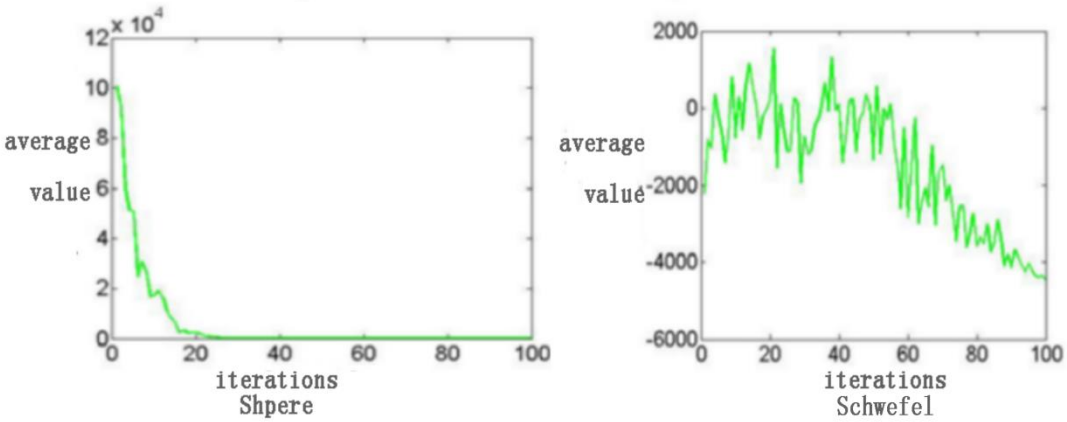

Figure 4.2: Average fit for Shere and Schwefel function tests

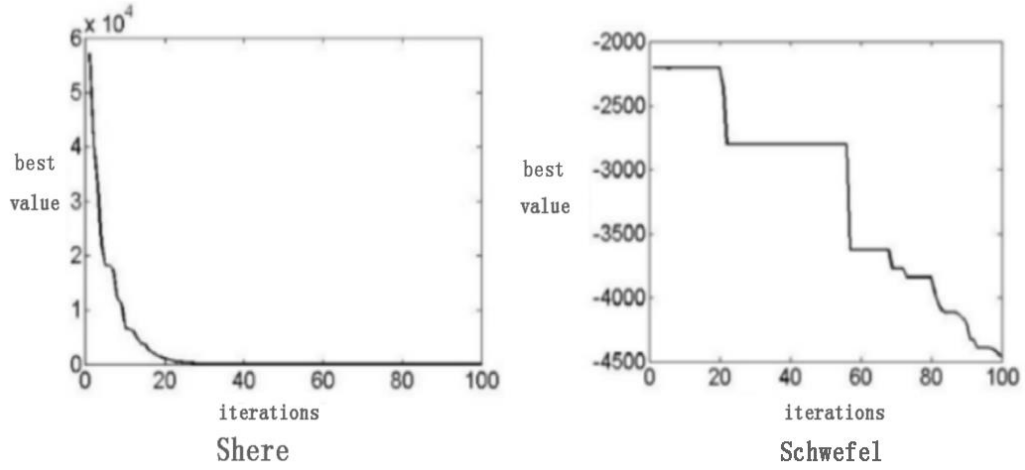

Figure 4.3: Optimal values for Shere and Schwefel function tests

Figure 4.4-4.6 shows the results of two test functions when the number of leaders is 2, including the search history of gray wolves in 100 iterations, change of parameter A in 100 iterations, trajectory of the first variable of the first Wolf in 100 iterations, average fitness value of all wolves in 100 iterations, and optimal value in each iteration.

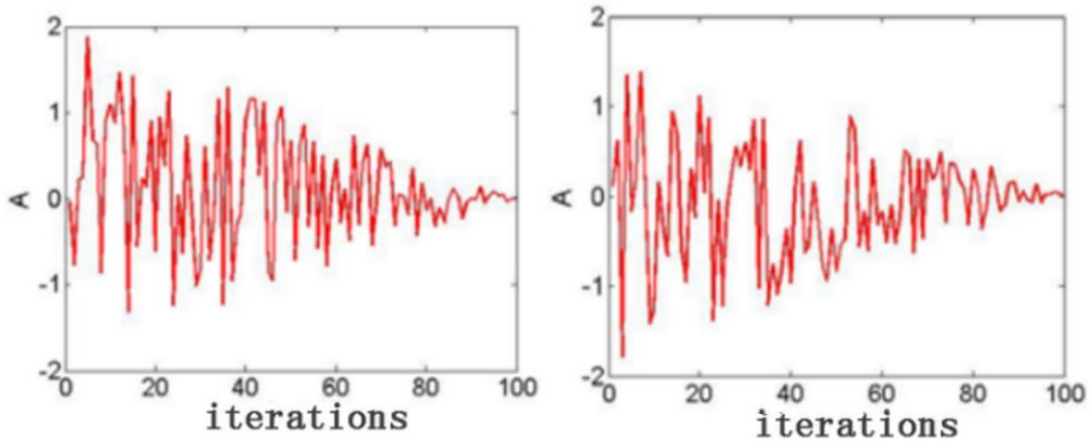

Figure 4.4 Test of Shpere and Schwefel Function A

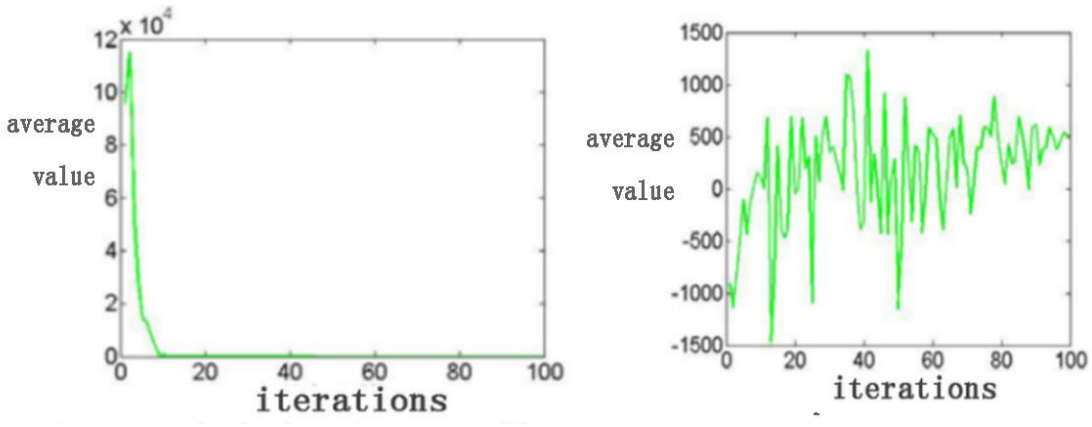

Figure 4.5: Average fit values for Shere and Schwefel function tests

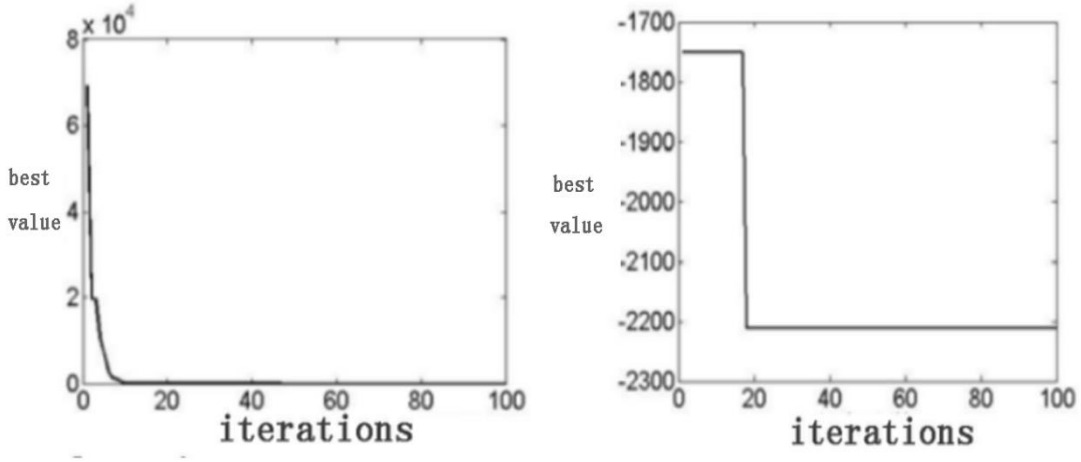

Figure 4.6: Optimal values for Shere and Schwefel function tests

Figure 4.7-4.9 shows the change in parameter A in 100 iterations when the number of leadership levels is 4, the trajectory of the first variable of the first Wolf in 100 iterations, the average fitness value of all wolves in 100 iterations, and the optimal value in each iteration.

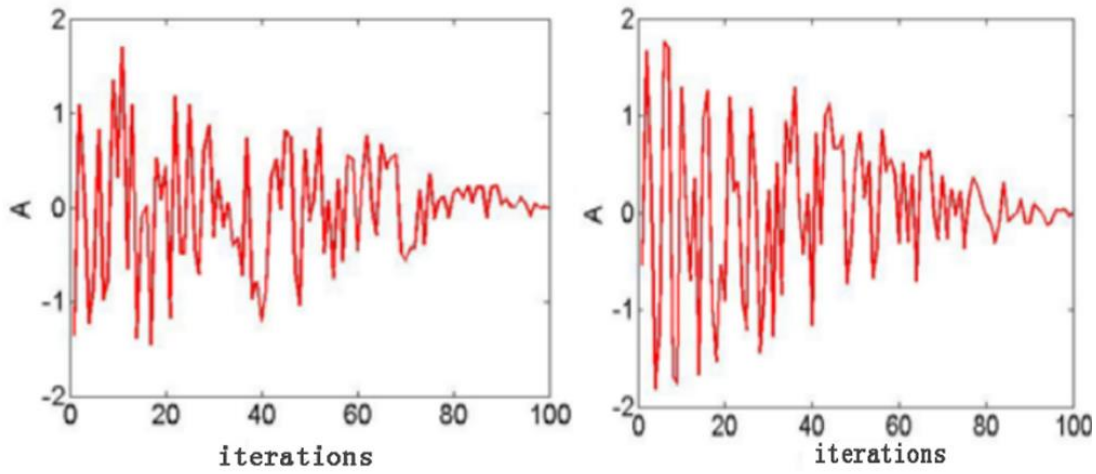

Figure 4.7 Test of Shpere and Schwefel Function A

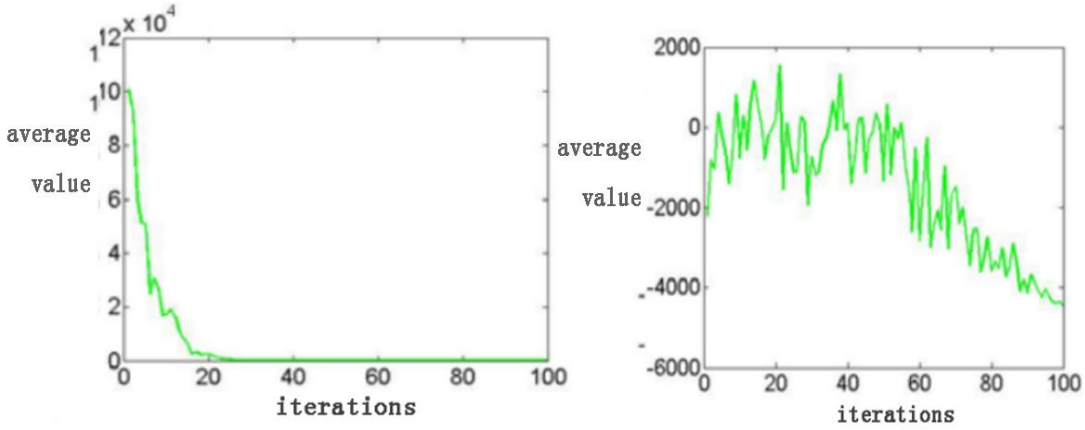

Figure 4.8: Average fit for Shere and Schwefel function tests

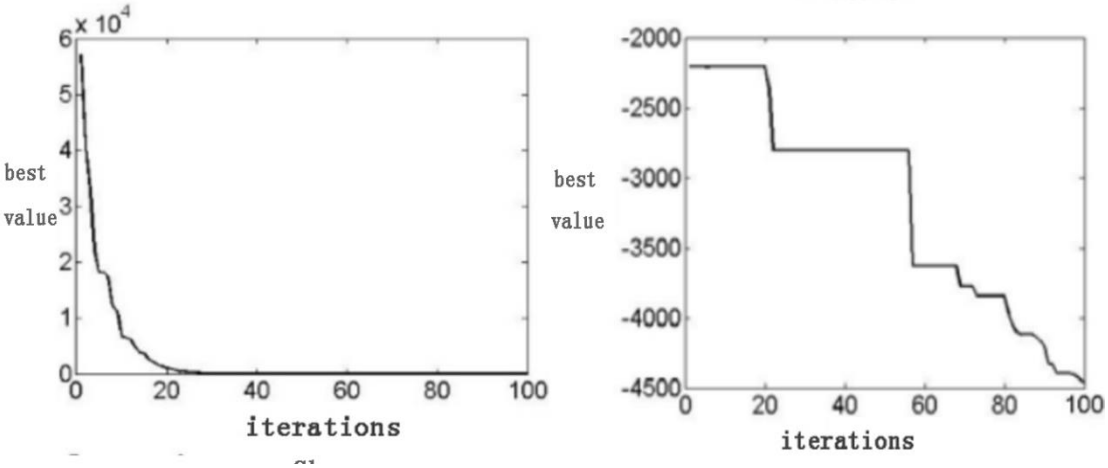

Figure 4.9: Optimal values for Shere and Schwefel function tests

4.3 Experimental analysis of Shpere and Schwefel functions

First of all, it can be seen from Figure 4.3 of the search history of the two functions that the solutions in GWO gradually approach the direction of the optimal solution in the evolution process, and eventually converge to the global optimal. Secondly, it can also be found in Figure 4.1 that the change of control parameter A plays a role in the change of the trajectory of the solution. In addition, in terms of the average fit value(Figure 4.2) and the best value(Figure 4.3), the average value of the global solution gradually decreases and shows an accelerated decline proportional to the number of iterations, and the global best value gradually stabilizes to the global optimal solution with the number of iterations.

The value of A directly affects the position update of leading wolves, so the larger the range of A, the larger the range of wolves' position movement and update, that is, the larger the search scope. Compared with Figure 4.4 and Figure 4.7, in terms of the search history of gray wolves in 100 iterations on Shpere and Schwefel functions, the PSO-EBGWO with the number of leaders at 4 has a wider and more diverse track distribution than that with the number of leaders at 2. This shows that the algorithm has a strong exploration ability when the number of leaders is large, but a relatively strong development ability when the number of leaders is small. In addition, as shown in Figure 4.6 and 4.9, the PSO-EBGWO of two leaders converges faster than that of four leaders. The experimental results show that the PSO-EBGWO algorithm can balance the exploration and development ability of the algorithm.

By comparing the experimental results of the GWO algorithm (leadership level 3) and PSO-EBGWO algorithm (leadership level 2 or 4), it can be seen from the best value in Figure 4.3 that the optimal value calculated when the number of iterations ranges from 0

to 20 and 20 to 60 is a straight line, which is easy to fall into the local optimal value. Of course, when the number of iterations reaches more than 70 times, the global optimal value is gradually obtained. Figure 4.3 Compared with Figure 4.6 and Figure 4.9 of the PSO-EBGWO algorithm, it can be seen from the optimal value that when the number of leaders is 2, the global optimal solution converges20 times, and the convergence time is very fast. When leadership level 4 is similar to leadership level 3, it is easy to fall into local optimal, but its exploration ability is relatively strong, which is stronger than the GWO algorithm. If the number of iterations of the algorithm is increased to more than 100 times, it will not fall into local optimal, but the required time is very long, which will seriously increase the cost of the enterprise, so the number of iterations can only be selected according to the actual situation.

To sum up, in the actual financial loan data, the amount of data is very large, the data sample is irregular, and different data segments of similar customers have similarities and differences. When the attributes of the financial loan data sample are easy to distinguish between good and bad customers, two leadership levels are required to improve the convergence speed of the algorithm and reduce the number of iterations. When the data is between the samples of good and bad customers, it is necessary to expand the search scope, and leadership level 4 is required to conduct exploration (4 leadership levels have strong exploration ability), and the number of iterations is correspondingly increased. Therefore, the PSO-EBGWO self-applicable leadership Grey Wolf optimization algorithm is reasonable, can effectively balance the exploration and development capabilities of the GWO algorithm, and effectively reduces the time complexity.

5. PSO-EBGWO-CatBoost experimental and analysis

The data sets selected for the experiment are the “LendingClub”, and then the experimental test and evaluation are carried out.

LendingClub is the world's largest peer-to-peer Internet lending platform. Its main business is to assess the default risk of borrowers and set different borrowing rates based on the borrower's past credit history and other information. Borrowers can quickly obtain loans by submitting applications. Investors gain interest or income by reviewing the borrower's past credit history and borrowing purpose and deciding whether to lend money to borrowers with different borrowing rates. The loan dataset collected online on 2.92 million Americans (one in 10 of the population) over 13 years, includes age, income, job title, geographic location, loan purpose, credit rating, and more. The Lending Club dataset can be used for our economic, demographic, social, job, and political data analysis and user profiling of Americans.

LendingClub 2020 has a total of more than 128,000 data, 110 variables. With a large amount of data and rich variables, the Lending club is an ideal data set for machine learning modelingvarious algorithm experiments (Misheva, Osterrieder, etc 2021)[21]. Financial research institutions can be used for data cleaning, variable screening, parameter adjustment, multi-algorithm comparison, unbalanced data processing, and other tests.

5.1 Experimental results of “LendingClub” data set

The model trained by CatBoost algorithm and PSO-EBGWO-CatBoost algorithm was tested, and then compared with CatBoost, LightGBM, XGBoost and logistic regression algorithms, and finally the model was evaluated.

With the same number of iterations, the experimental time of PSO-EBGWO-CatBoost, LightGBM、XGBoost、CatBoost and Logistic regression is shown in Figure 5.1.

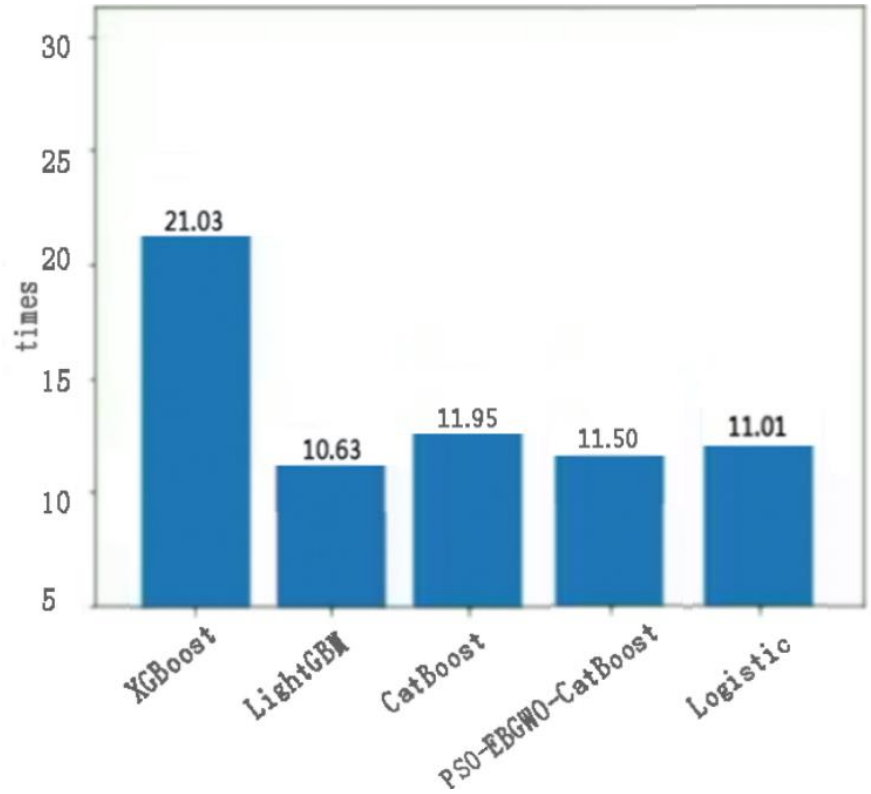

Figure 5.2: Time profile of different algorithms running

The average values of each index of the CatBoost algorithm and PSO-EBGWO-CatBoost algorithm were iterated 50 times, 100 times, 200 times, and 500 times respectively during the experiment, as shown in Table 5.1.

Table 5.1: Model validation indicator results

CatBoost time	Accuracy	Recall	AUC	PSO-EBGWO-CatBoost time	Accuracy	Recall	AUC
50	85.97	84.42	85.02	50	93.84	92.90	92.05
100	89.54	85.37	86.89	100	93.82	92.88	92.10
200	89.60	85.41	86.90	200	93.87	92.91	92.04
500	89.57	85.39	86.85	500	93.86	92.89	92.07

Here, the two algorithms "PSO-EBGWO-CatBoost", "CatBoost", ”XGBoost”with higher accuracy are selected to compare the ROC curves, as shown in the figure 5.3.

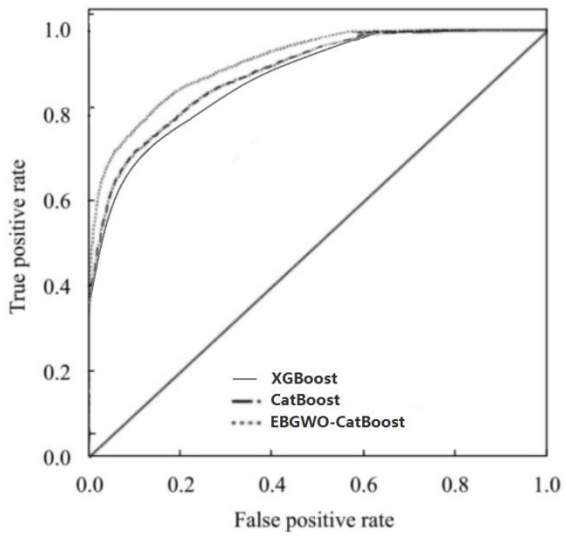

Figure 5.3: Model ROC Curve

In order to verify the accuracy of the PSO-EBGWO-CatBoost model, the CatBoost, Logistic, LightGMB and XGBoost models before parameter optimization were introduced in the experiment for training, and the accuracy of the test set is shown in Table 5.2.

Table 5.2: Model Performance evaluation Table (unit: %)

Model	Accuracy	Precision	Recall	AUC	F1	Feature number
PSO-EBGWO-CatBoost	93.84	92.90	91.53	92.05	0.541	99
CatBoost	89.54	88.01	85.37	86.89	0.503	110
XGBoost	89.31	88.03	87.49	86.37	0.497	110
LightGMB	73.06	80.05	79.98	83.01	0.455	110
Logistic	73.07	770.88	72.00	74.34	0.467	110

5.2 Experimental discuss of “LendingClub” data set

Result analysis: From the analysis of time complexity,

Figure 5.3 illustrates that the time difference between the PSO-EBGWO-CatBoost combination algorithm and the CatBoost algorithm is negligible. Under the same number of iterations, the PSO-EBGWO-CatBoost algorithm takes 0.55 seconds less than the CatBoost algorithm, and only 0.49 seconds longer than the traditional logistic regression algorithm. The new algorithm is almost the same as the traditional logistic regression algorithm, so the new algorithm can replace the old algorithm in running time.

The analysis in Table 5.1 shows that the fast convergence ability of PSO-EBGWO-CatBoost algorithm is stronger than that of CatBoost algorithm. From the analysis of Table 5.2 and Figure 5.3, it is also proved that the PSO-EBGWO-CatBoost combination algorithm has the best comprehensive performance and can replace the algorithm of the traditional risk control model.

6. CONCLUSION

Aiming at the problem of loan risk prediction of financial institutions, this paper has done a series of work from the aspects of data sampling, feature engineering, and classification

algorithm through the vehicle mortgage loan data of a domestic financial company, and drawn the following conclusions:

(1) Compared with the traditional classification model, the PSO-EBGWO-CatBoost model has improved the accuracy, error rate, recall rate, and AUC curve, and has an excellent minority class recognition rate in the face of unbalanced data sets, which has a high application value.

(2) As a branch of the gray Wolf optimization algorithm, PSO-EBGWO can avoid the problem that the Gray Wolf Optimization algorithm is easy to fall into local convergence by improving the example learning method to a certain extent and improving the work efficiency. In this experimental scenario, the PSO-EBGWO algorithm is superior to GWO. The PSO-EBGWO-CatBoost model can effectively improve the model's accuracy. In conclusion, compared with XGBoost, LightGMB, XGBoost,and other commonly used credit risk assessment methods, the PSO-EBGWO-CatBoost model can predict loan risk more effectively.

References