Hindawi
Discrete Dynamics in Nature and Society
Volume 2021, Article ID 5080472, 13 pages
https://doi.org/10.1155/2021/5080472

Research Article

Research on Default Prediction for Credit Card Users Based on XGBoost-LSTM Model

Jing Gao,[1,2] Wenjun Sun,[1] and Xin Sui[2,3]

[1]*School of Management, Harbin Institute of Technology, Harbin 150006, China*
[2]*Department of Risk Management, Harbin Bank, Harbin 150010, China*
[3]*School of Finance, Harbin University of Commerce, Harbin 150028, China*

Correspondence should be addressed to Jing Gao; gaojing2@hrbb.com.cn

Received 11 October 2021; Revised 22 November 2021; Accepted 29 November 2021; Published 16 December 2021

Academic Editor: Ahmed Farouk

The credit card business has become an indispensable financial service for commercial banks. With the development of credit card business, commercial banks have achieved outstanding results in maintaining existing customers, tapping potential customers, and market share. During credit card operations, massive amounts of data in multiple dimensions—including basic customer information; billing, installment, and repayment information; transaction flows; and overdue records—are generated. Compared with preloan and postloan links, user default prediction of the on-loan link has a huge scale of data, which makes it difficult to identify signs of risk. With the recent growing maturity and practicality of technologies such as big data analysis and artificial intelligence, it has become possible to further mine and analyze massive amounts of transaction data. This study mined and analyzed the transaction flow data that best reflected customer behavior. XGBoost, which is widely used in financial classification models, and Long-Short Term Memory (LSTM), which is widely used in time-series information, were selected for comparative research. The accuracy of the XGBoost model depends on the degree of expertise in feature extraction, while the LSTM algorithm can achieve higher accuracy without feature extraction. The resulting XGBoost-LSTM model showed good classification performance in default prediction. The results of this study can provide a reference for the application of deep learning algorithms in the field of finance.

1. Introduction

Both the issuance of credit cards and the scale of credit have increased steadily in recent years. According to data from the People's Bank of China, at the end of 2020, the number of credit cards issued totaled 778 million, and the credit balance of credit cards was 7.91 trillion yuan. With the effects of the COVID-19 pandemic, the quality of credit card assets deteriorated in 2020, and indicators such as overdue scale and nonperforming indicators are trending upward in the short term. The total overdue credit for the last half year was 83.8 billion yuan, accounting for 1.06% of the outstanding credit card balance, which marked an increase of 0.08% from the end of 2019. On the premise of ensuring the stable development of the credit card business, determining the effective management of credit card customers and reducing the problems caused by customer default have become a focus of attention. Effective use of already-collected customer information to identify customers who may default is a key measure to increase profits. Based on the results of default predictions, banks can reduce or freeze the credit lines of accounts that may default, thereby reducing their risk exposure. This would prevent an increase in the balance of an account destined to default and save financial institutions hundreds of millions of yuan in losses each year.

Current default prediction is primarily based on account, credit bureau, and transaction flow data. The account-level data include, among other information, month-end balance, credit limit, borrower income, account activity, and arrears. Credit bureau data include credit score, total credit limit, total outstanding balance on all cards, and the number of outstanding accounts. Transaction flow data contain a large amount of information related to default prediction, such as consumption, installment, and repayment habits. Transaction flow data is easy for financial institutions to obtain, is difficult to forge, and has a high degree of authenticity. Khandani et al. [1] integrated account credit records, financial behavior, and user-level transaction flow data to predict overdue payments; the total transaction amount of nearly 20 categories was extracted as the feature value for the flow data. Their research indicated that any forward-looking insights about consumer credit collected from historical consumer behavior data are crucial. Machine-learning predictions are highly adaptable and can capture the dynamics of the ever-changing credit cycle and the absolute level of default rates.

With the widespread application of artificial intelligence technology, scholars are increasingly applying machines to research on default prediction [2–8]. Butaru et al. [9] collected customer credit card data from multiple banks. These data included internal account-level information from banks and consumer data from large credit bureaus in the United States. Three data mining models—decision tree, random forest, and logistic regression—were used to study customer default predictions; the comparison showed that the decision tree and random forest models were better than logistic regression in credit card prediction accuracy. Since Chen and Guestrin [10] proposed the XGBoost algorithm in 2016, it has been used in many fields, including disease diagnosis, image recognition, and personal credit. Studies have shown that XGBoost can provide better prediction accuracy than other methods. Zhang and Chen [11] applied XGBoost to bond default risk prediction and found that the XGBoost algorithm was superior to traditional algorithms (e.g., LR, SVM, and KNN) for dealing with imbalanced data. Given its high performance in various fields, this study used the XGBoost algorithm as a representative machine-learning algorithm to establish a default prediction model and used it as a benchmark for comparison with deep learning models.

The default prediction model based on machine learning requires extracting features of the transaction flow data. The quality of the model largely depends on the application of feature engineering [12–14]. The LSTM [15] algorithm performs well in time-series data mining and has had many applications in the financial field [16–19], such as customer service marketing, risk control, and trading strategy. In the field of antifraud, for example, deep learning technology automatically recognizes fraudulent transactions from massive amounts of transaction data, realizes successful interception, and blocks fraudulent transactions, thereby improving system effectiveness, reducing the rate of false alarms, and reducing compliance risks [20]. This article focuses on exploring the application of deep learning technology in processing transaction flow data to improve the accuracy of default prediction. Machine learning algorithms and deep learning algorithms are used on the same data set to construct default prediction models, and the prediction accuracy and modeling workload are compared, ultimately revealing that the deep learning model has high prediction accuracy and does not require features for default prediction. The results of this study have practical significance for guiding financial institutions in reducing losses caused by credit card customers who default.

2. Data

The source used to establish the default prediction model is user data from a certain small-scale commercial bank's credit card services. The use of the data is authorized by the bank, and the data is desensitized and does not contain the user's identity information. Most cardholders are small business owners, and the consumption amount exceeds 500 yuan automatically in installments. Thus, most users use IOUs or cash withdrawals, and there are only a few outgoing transactions and repayment transactions every month.

2.1. Data Type Description.

2.1. Data Type Description. The data include basic user information and installment, billing, transaction flow, and credit information; the variables included in these are all monthly variables. In addition to directly using the monthly values within the observation period, statistical information is also used, including the maximum, minimum, and average values, as well as the ratio of the amount to the credit limit during the period.

2.1.1. Basic Information. Basic information includes not only information such as age, gender, and marital status, but also customer hierarchical code, which is a comprehensive indicator based on the user's occupation; company nature and size; position; and annual income. It also includes the number of days since the card was created to the first use of the credit card processed according to user behavior, which can reflect the user's desire for funds.

2.1.2. Billing, Installment, and Credit Information. Billing information describes the user's past bill amount and overdue status. Installment information is the number and amount of different installment states (e.g., new, activated, completed in advance, and completed), as well as the main installment type and expiration time. The credit information is the PBOC credit score, which is updated monthly.

2.1.3. Transaction Flow Information. Transaction flow data contain the user ID and transaction date, time, type, and amount. The transaction type is a 4-digit code, covering more than 100 transaction types; the transaction amount is in RMB, and positive means outgoing, while negative means incoming. As shown in Table 1, the length of monthly transaction data differs by cardholder, which is typical of unstructured data. With the large amount of data, this forms the main difficulty for processing transaction data.

2.2. User-Defined. A good user is defined as one having no overdue situation within one year after the observation point. Bad users are defined as users who have been overdue for more than 60 days within one year after the observation point. Users who are overdue for 0–60 days are uncertain users and were not adopted. The total number of samples is

Table 1: Example of transactions flow data.

User ID	Input date	Input time	Transaction type	Bill amount	Credit limit
100001	20170605	15:25	4110	48.28	50000
100001	20170605	15:25	2110	5000.00	50000
100001	20170604	10:51	7098	−10500.00	50000
100001	20170605	15:26	4110	50.00	50000
100001	20170610	23:58	5000	148.09	50000
100001	20170605	15:26	2110	5000.00	50000
100002	20170605	21:05	7038	−600.00	100000
100002	20170610	20:04	3254	1116.00	100000
100003	20170610	20:04	3254	1813.50	20000
100004	20170605	14:01	2000	6600.00	50000
100004	20170610	20:04	3250	80.00	50000
100004	20170610	20:04	3250	25.60	50000
...	...	...	...	...	...

140,000, of which 80% are randomly selected as training samples and 20% as test samples. In both training data and test data, bad customers accounted for 20% of the sample.

3. Model Evaluation Index

The confusion matrix and area under the curve (AUC) were used to evaluate the models. The confusion matrix summarizes the records in the data set in the form of a matrix according to the two criteria of the real category and the classification judgment made by the classification model [21]. The name is derived from the fact that it can easily be indicated whether there is confusion among multiple categories (i.e., a positive category is predicted as a negative category). P and N represent the positive and negative judgment results of the model, respectively. T and F represent the judgment results of the model as True and False. The confusion matrix is defined as follows:

$$\text{confusion matrix} = \begin{bmatrix} \text{TP} & \text{FN} \\ \text{FP} & \text{TN} \end{bmatrix}. \tag{1}$$

Knowing the confusion matrix, the accuracy rate (ACC), precision, and recall can be calculated, and the formulas are as follows:

$$\begin{aligned} \text{ACC} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}, \\ \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}. \end{aligned} \tag{2}$$

The ACC value is calculated based on the cutoff value between positive and negative examples. Compared with the ACC value, the AUC value can integrate the predictive performance of all cutoff values [11, 22]. AUC represents the area under the receiver operating characteristic (ROC) curve, which is between 0 and 1. As a value, AUC can intuitively evaluate the quality of the classifier: the larger the value, the better. It is calculated as follows:

$$\text{AUC} = \frac{\sum_{\text{positive}} \text{rank}_i - (M(M+1)/2)}{M * N}. \tag{3}$$

4. XGBoost Model

4.1. XGBoost Algorithm. Extreme gradient boosting (XGBoost) is a boosting-type tree algorithm proposed by Chen and Guestrin. It is widely used in web text and product classification, as well as customer behavior prediction, and it has achieved the most advanced results in many machine learning competitions [10]. Its wide application benefits from optimization in the following four aspects: a distributed weighted square graph algorithm that solves the problem of segmentation point selection, better processing of sparse data, efficient cache-aware block data storage structure, and better usage of parallel and distributed computing.

The prediction results of the model consisting of K decision trees are

$$\hat{y}_i = \phi(\mathbf{x}_i) = \sum_{k=1}^{K} f_k(\mathbf{x}_i), \quad f_k \in \mathcal{F}, \tag{4}$$

where $\mathbf{x}_i$ is the i-th input sample; $\hat{y}_i$ is the predicted value calculated through the mapping relationship $\mathbf{f}_k$; and $\mathcal{F}$ is the collection of mapping relationships. The optimization objective and loss function are defined as

$$\begin{aligned} \mathcal{L}(\phi) &= \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \\ \Omega(f) &= \gamma T + \frac{1}{2}\lambda \|\omega\|^2. \end{aligned} \tag{5}$$

Here, $\mathbf{l}$ is the differentiable convex loss function, which measures the difference between the predicted value $\hat{y}_i$ and the target value $\mathbf{y}_i$, and Ω is an additional regularization term to help smooth the weight of the model to avoid overfitting.

The above test contains functions as parameters, which cannot be optimized in Euclidean space using traditional methods; $\hat{y}_i^{(t)}$ is defined as the i-th prediction of the t-th iteration. Then, the loss function is defined as

$$\mathscr{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)\right) + \Omega(f_t). \tag{6}$$

A second-order Taylor expansion is performed on the previous equation:

$$\mathscr{L}^{(t)} \cong \sum_{i=1}^{n}\left[l\left(y_i, \hat{y}_i^{(t-1)}\right) + g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t). \tag{7}$$

The constant term is removed to yield the simplified objective function of the t-th iteration:

$$\begin{aligned}\tilde{\mathscr{L}}^{(t)} &= \sum_{i=1}^{n}\left[g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t),\\ g_i &= \partial_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right),\\ h_i &= \partial^2_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right),\end{aligned} \tag{8}$$

where $\mathbf{g}_i$ is the first-order partial derivative of $\mathbf{l}(\mathbf{y}_i, \hat{y}_i^{(t-1)})$ to $\hat{y}_i^{(t-1)}$ and h_i is the second-order partial derivative of $\mathbf{l}(\mathbf{y}_i, \hat{y}_i^{(t-1)})$ to $\hat{y}_i^{(t-1)}$. $\mathbf{I}_j$ is defined as the sample set on each leaf node, and ω_j as the weight of the corresponding leaf node. Formula (8) can be rewritten as

$$\begin{aligned}\tilde{\mathscr{L}}^{(t)} &= \sum_{i=1}^{n}\left[g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T}\omega_j^2,\\ \tilde{\mathscr{L}}^{(t)} &= \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i\right)\omega_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right)\omega_j^2\right] + \gamma T.\end{aligned} \tag{9}$$

For a certain tree structure $\mathbf{q}(\mathbf{x})$, the optimal weight ω_j^* for the leaf nodes can be calculated by the following formula:

$$\omega_j^* = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda}. \tag{10}$$

The corresponding optimal objective function can be calculated as follows:

$$\tilde{\mathscr{L}}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T. \tag{11}$$

4.2. Transaction Flow Feature Extraction. Using the XGBoost model for overdue forecasting requires feature extraction of the original transaction data and conversion of original data of unequal length into feature data of equal length. The transaction data was first classified and counted based on business experience and historical documents. The process of feature extraction is to initially screen out important features based on business experience and then derive feature statistics. First, the data was divided into four main categories according to transaction type: consumption, cash withdrawal, fee, and repayment. The number of transactions was then separately counted, and the total transaction amount for each major category was found. Finally, the statistics were calculated for three months. In the first stage, 48 feature values were extracted, as shown in Table 2. However, the prediction results obtained from these features are not ideal. In the second stage, we continued to dig into the data to understand the business background and added 34 features (totaling 82 features, as shown in Table 2). The added features include the number of days for repayment in advance, statistics on the number of transactions per month, and separate statistics for a special transaction type: penalty.

4.3. XGBoost Model Establishment. The flowchart for XGBoost model establishment is shown in Figure 1. First, user-related basic information, credit information, and billing, installment, and transaction flow were collected and preprocessed (e.g., data screening, classification label generation, and missing data supplementation). Then, the flow characteristics of the transaction flow data were extracted and input into the XGBoost classification model. For other types of data, monthly data generation and statistical calculation were performed for the observation period and then entered into the XGBoost classification model as feature values. Finally, the XGBoost classification model was trained. If the model evaluation conditions are met, the establishment of the default prediction model is complete.

4.4. Feature Importance. Figure 2 shows the top 20 most important features and their weights for the XGBoost model. The default classification is primarily based on penalty-related features and the PBOC score. Five of the top 20 most important indicators are related to transaction flow data, and the importance accounts for a total of 24%. This shows that the transaction flow data contain important information related to default prediction. Penalty-related features are not in the preliminary extraction of the transaction flow data but are important features selected after in-depth analysis of each transaction type. The model evaluation indicators of XGBoost algorithm models using different features are shown in Table 3. When the transaction features are not used for overdue prediction, the AUC value of the XGBoost model is 71%. The prediction effect of using 48 preliminary extracted transaction features in the first stage is better than not using them. Compared with using preliminary extracted transaction features, the AUC is increased by 13%, and the recall rate is increased by 34% when using all 82 transaction features, which is a significant improvement, indicating that the features mined in the second stage have better distinguishing ability. Feature extraction is thus the most important and most arduous step in machine learning modeling.

4.5. XGBoost Model Evaluation. The confusion matrix of the test data based on the XGBoost model is shown in Figure 3(a). According to the confusion matrix, the ACC is 86.5%, the precision is 74.1%, and the recall is 51.5%. The

Table 2: Extracted features from transaction data.

Transaction type	First month	Second month	Third month	Max	Min	Mean	Others
Amount of consumption	c_sum_1	c_sum_2	c_sum_3	c_sum_max	c_sum_min	c_sum_mean	c_sum_max/credit
Number of consumption	c_num_1	c_num_2	c_num_3	c_num_max	c_num_min	c_num_mean	
Amount of fees	f_sum_1	f_sum_2	f_sum_3	f_sum_max	f_sum_min	f_sum_mean	f_sum_max/credit
Number of fees	f_num_1	f_num_2	f_num_3	f_num_max	f_num_min	f_num_mean	
Amount of repayments	r_sum_1	r_sum_2	r_sum_3	r_sum_max	r_sum_min	r_sum_mean	r_sum_max/credit
Number of repayments	r_num_1	r_num_2	r_num_3	r_num_max	r_num_min	r_num_mean	
Amount of cash withdrawals	cw_sum_1	cw_sum_2	cw_sum_3	cw_sum_max	cw_sum_min	cw_sum_mean	cw_sum_max/credit
Number of cash withdrawals	cw_num_1	cw_num_2	cw_num_3	cw_num_max	cw_num_min	cw_num_mean	
Amount of penalty	penalty_sum_1	penalty_sum_2	penalty_sum_3	penalty_sum_max	penalty_sum_min	penalty_sum_median	penalty_sum_max/ credit
Number of penalty	penalty_num_1	penalty_num_2	penalty_num_3	penalty_num_max	penalty_num_min	penalty_num	
Repayment days in advance	pre_pay_days_1	pre_pay_days_2	pre_pay_days_3	pre_pay_days_max	pre_pay_days_min	pre_pay_days_median	
Number of transaction flow	event_counts_1	event_counts_2	event_counts_3	event_counts_max	event_counts_min	event_counts_median	event_counts_sum
Main transaction types	trans_type_mode_1	trans_type_mode_2	trans_type_mode_3				trans_type_mode_all

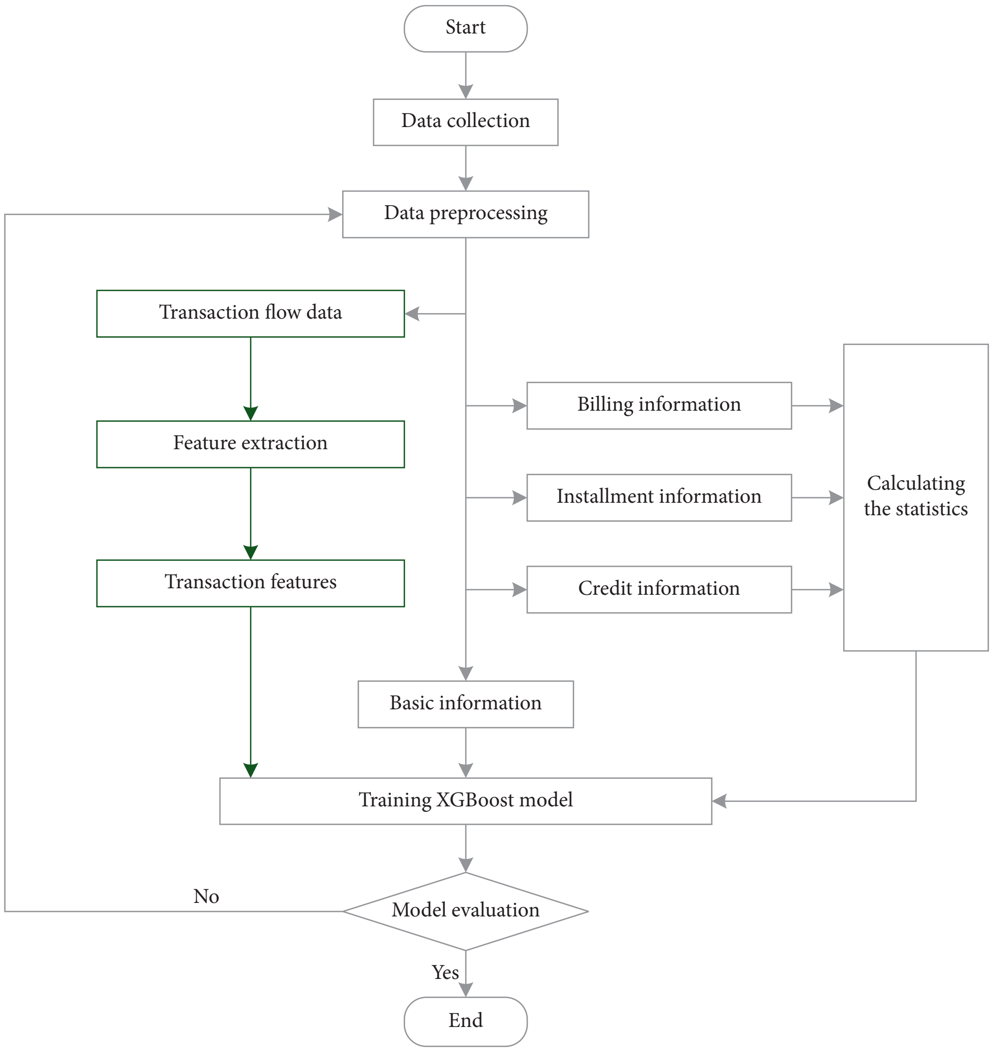

FIGURE 1: The flow chart of XGBoost model establishment.

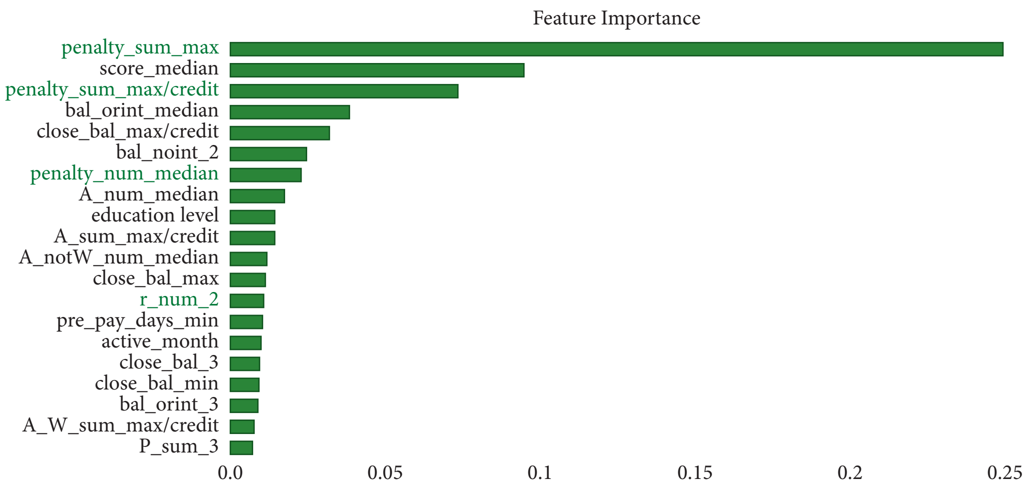

FIGURE 2: Feature importance index ranking of the XGBoost model.

Table 3: Model evaluation indicators of XGBoost algorithm models using different features.

Features	Train AUC	Train ACC	Test AUC	Test ACC	Test recall	Test precision
Without transaction features	0.716	0.801	0.710	0.799	0.008	0.493
The first stage	0.764	0.821	0.759	0.819	0.163	0.714
The first and second stages	0.895	0.873	0.893	0.870	0.507	0.768

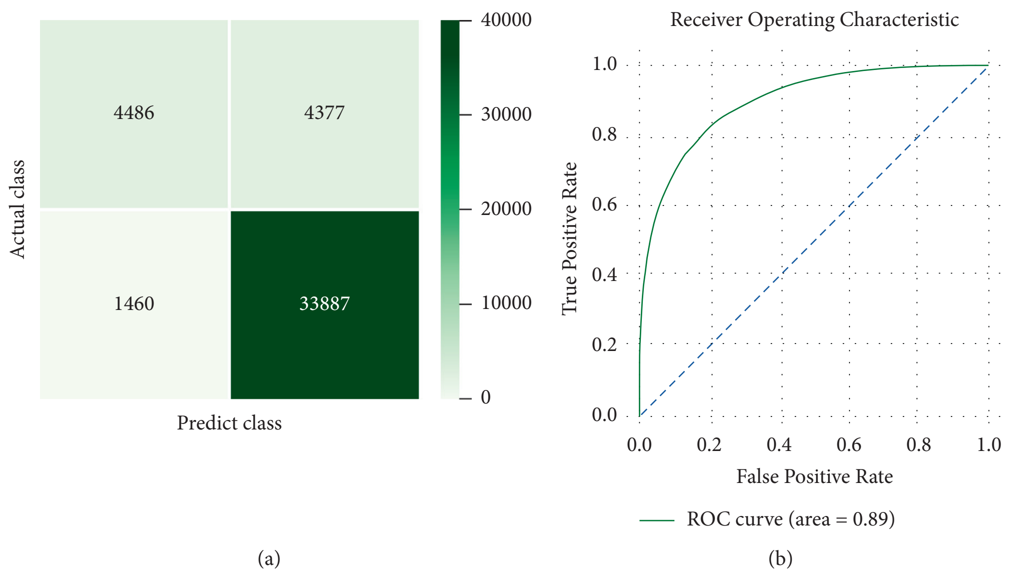

Figure 3: The confusion matrix and ROC of the XGBoost model. (a) The confusion matrix of the test data. (b) The ROC curve of the test data.

ROC curve of the test data is shown in Figure 3(b), and the AUC is 89%.

4.6. Comparison of Machine Learning Models. The corresponding data packages in Python were used to train the machine learning models for default prediction. Model parameter settings and data preprocessing methods are shown in Table 4. Except for the necessary normalization of the K neighborhood and SVM algorithm, the other algorithms used the same original data. The principle of parameter setting was to set as few parameters as possible without overfitting.

The model evaluation indicators of machine learning algorithms are shown in Table 5. The K neighborhood and SVM algorithm for default prediction appear to be less effective, and the decision tree, random forest, AdaBoost, and XGBoost algorithms are better; of these, the XGBoost algorithm is the best. Figure 4 shows the ROC curves for the different algorithms.

5. XGBoost-LSTM Model

5.1. LSTM Algorithm. The traditional neural network model is fully connected from the input layer to the hidden layer and then to the output layer, which means there is no connection between nodes in the same layer, and the propagation of the network is sequential. This kind of network structure often appears powerless to deal with sequence or time-series problems because of its lack of memory. Therefore, a new kind of network—recurrent neural network (RNN)—is required. LSTM is a type of RNN that is especially good at processing sequence data. The ingenuity of LSTM is that, by increasing the input, forgetting, and output thresholds, the weight of the self-loop is changed. In the case of fixed model parameters, the integration scale at different times can be dynamically changed, thereby avoiding the problem of gradient disappearance or expansion. Figure 5 illustrates the structure of the LSTM unit.

5.1.1. Input Gate. The input gate ($g_i^{(t)}$) controls the information entered into the internal storage unit, which can be expressed as follows:

$$g_i^{(t)} = \sigma \cdot \left(b_i^g + \sum_j U_{ij}^g x_j^{(t)} + \sum_j W_{ij}^g h_j^{(t-1)} \right). \tag{12}$$

σ is the sigmoid function; $x^{(t)}$ is input vector at time t; $h^{(t)}$ is hidden layer vector, including the output of all LSTM units; and b^g, U^g, and W^g represent the deviation, the input weight, and the cycle weight of the input gate, respectively.

5.1.2. Forget Gate. The forget gate ($f_i^{(t)}$) controls how much information from the previous time is stored in the internal storage unit, which can be expressed as

Table 4: Machine learning model data processing methods and model settings.

Algorithm	Data processing	Model settings
K neighbors	Normalization, missing value supplement	n_neighbors = 10
SVM	Normalization, missing value supplement	kernel = “rbf,” max_iter = 500
DecisionTree	—	max_depth = 4
RandomForest	—	max_depth = 4
AdaBoost	—	max_depth = 4
XGBoost	—	max_depth = 4

Table 5: Model evaluation indicators of machine learning algorithms.

Algorithm	Train AUC	Train ACC	Test AUC	Test ACC	Test recall	Test precision
K neighbors	0.844	0.836	0.547	0.794	0.017	0.274
SVM	0.714	0.793	0.738	0.772	0.455	0.435
DecisionTree	0.855	0.855	0.852	0.853	0.459	0.705
RandomForest	0.857	0.841	0.854	0.838	0.245	0.818
AdaBoost	0.890	0.866	0.884	0.863	0.507	0.725
XGBoost	0.920	0.916	0.889	0.865	0.515	0.734

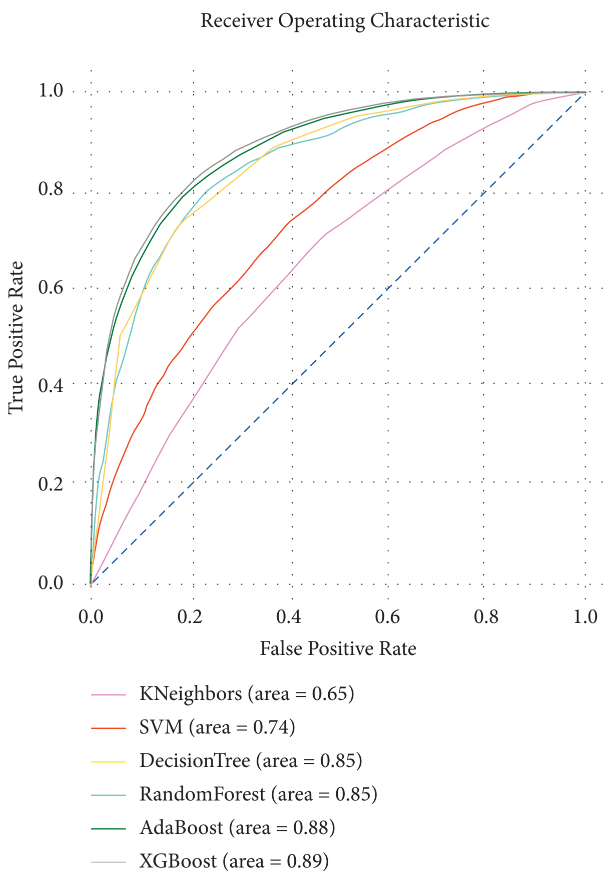

Figure 4: The ROC curve of machine learning algorithms.

$$f_i^{(t)} = \sigma \cdot \left(b_i^f + \sum_j U_{ij}^f x_j^{(t)} + \sum_j W_{ij}^f h_j^{(t-1)} \right). \tag{13}$$

In formula (13), b^f, U^f, and W^f represent, respectively, the deviation, input weight, and loop weight of the forget

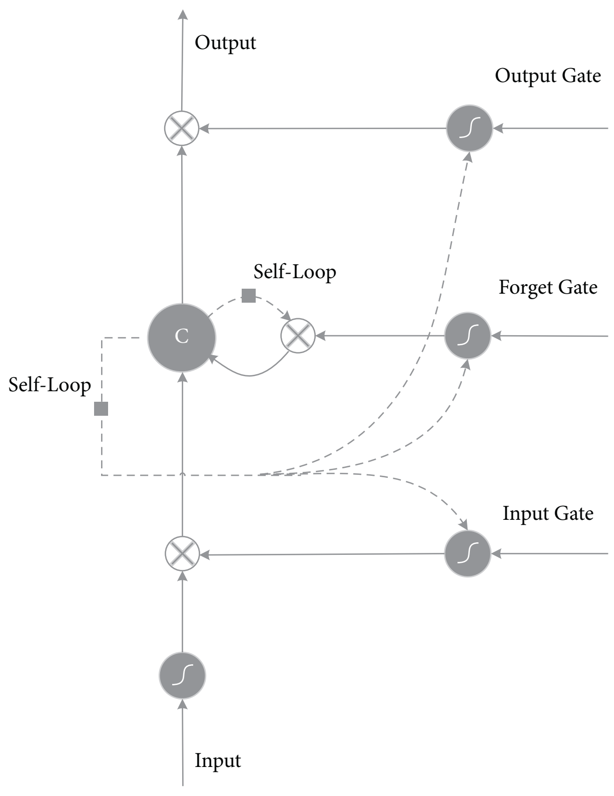

Figure 5: The structure of the LSTM unit.

gate. The expression of the internal storage unit of the LSTM unit is as follows:

$$S_i^{(t)} = f_i^{(t)} S_i^{(t-1)} + g_i^{(t)} \sigma \cdot \left(b_i + \sum_j U_{ij} x_j^{(t)} + \sum_j W_{ij} h_j^{(t-1)} \right), \tag{14}$$

where b, U, and W represent the deviation, input weight, and cycle weight of the LSTM unit, respectively. On the right side

Table 6: Example of LSTM model input data example.

No.	Days before billing	Input time	Transaction type	Bill amount	Bill amount/credit limit ∗ 100
1	30	20	7028	−200	0
2	18	7	7028	−1013	−3
3	10	0	7028	−9.45	0
4	10	0	3254	9.45	0
5	10	0	2020	1000	3
6	0	13	7028	−95.67	0
7	0	20	3254	125.55	0
8	0	20	3254	36.27	0
9	0	0	0	0	0
...	...	...	...	...	...
31	0	17	7028	−156	0
32	0	20	3254	121.5	0
33	0	20	3254	35.1	0
34	0	20	3254	27.41	0
35	0	20	3254	24.3	0
36	0	0	0	0	0
37	0	0	0	0	0
38	0	0	0	0	0
39	0	0	0	0	0
...	...	...	...	...	...
61	20	16	7028	−1150	−3
62	20	16	7028	−1480	−4
63	20	16	3254	12.87	0
64	20	16	2020	2600	8
65	19	16	7028	−8000	−26
66	19	17	7028	−1045	−3
67	19	17	3254	48.6	0
68	19	17	2020	9000	30
69	0	14	7028	−300	−1
...	...	...	...	...	...

of formula (14), the first half is the cell state information controlled by the forget gate, and the second half is the input information controlled by the input gate [16].

5.1.3. Output Gate. The output gate ($O_i^{(t)}$) controls the internal storage unit by releasing and generating the required information, which can be given by the following formula:

$$O_i^{(t)} = \sigma \cdot \left(b_i^o + \sum_j U_{ij}^o x_j^{(t)} + \sum_j W_{ij}^o h_j^{(t-1)} \right),$$
$$h_i^{(t)} = \tanh\left(S_i^{(t)}\right) O_i^{(t)}. \tag{15}$$

Here, b^o, U^o, and W^o represent the deviation, input weight, and loop weight, respectively, of the output gate.

5.2. Transaction Flow Data Processing. The establishment of a default prediction model using the LSTM algorithm does not require feature extraction; the data processing work mainly involves interception and complementation. According to the distribution of the average number of transactions per month by user, 30 was selected as the threshold for the number of transactions per month. Data exceeding the threshold were discarded, and insufficient data were filled with zeros. The 3 months of transaction flow data were then spliced. Table 6 shows the input data for the LSTM model of a certain account after processing.

5.3. XGBoost Model Establishment. The flowchart for establishing the XGBoost-LSTM model is shown in Figure 6. The processing of transaction flow data is different from the process of establishing the XGBoost model. There is no need to perform feature extraction for the transaction flow data; directly train the LSTM flow data classification model on the spliced flow data during the observation period. The classification results of the LSTM model were used as features input into the XGBoost model for training.

5.4. Feature Importance. Figure 7 shows the top 20 most important features and their weights for the XGBoost-LSTM model. The default classification of the XGBoost-LSTM model was based on outputs of LSTM model and the PBOC score.

5.5. XGBoost-LSTM Model Evaluation. The confusion matrix for the test data based on the XGBoost-LSTM model is shown in Figure 8(a). The ACC is 93.6%, the precision is 92.8%, and the recall is 73.6%. The ROC curve of the test data is shown in Figure 8(b), and the AUC is 0.95.

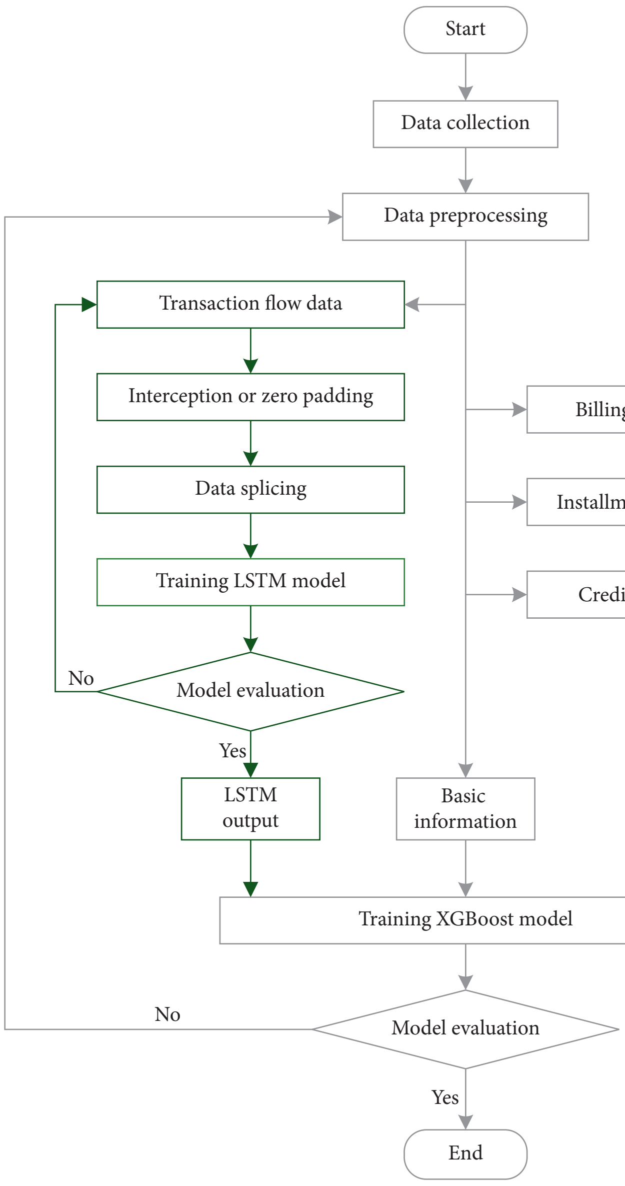

Figure 6: The flow chart of the XGBoost-LSTM model establishment.

5.6. Comparison of XGBoost Model and XGBoost-LSTM Model

5.6.1. Data Processing. Table 7 shows the data sources and data processing methods required by different algorithms. The XGBoost-LSTM model requires the least amount of work, which reduces the work of feature extraction from the transaction flow data.

5.6.2. Model Performance. The model evaluation indicators are shown in Table 8. Combining the data in Table 3, when the transaction features are not used for overdue prediction, the test set AUC of the model is 71%, the recall rate is 0.8%, and the accuracy is 49%. Adding the two output results of the LSTM model, the AUC of the model test set increased by 24%, the recall rate increased by 73%, and the accuracy rate increased by 43%. The significant increase in evaluation indicators proves the usefulness of the output features of the LSTM model. At the same time, the XGBoost-LSTM model has less improvement in AUC and accuracy compared with the XGBoost model building with the extracted transaction features. This shows that, under the premise of extracting useful features, the XGBoost model can also have good predictive performance. However, the extraction of useful features is based on deep business experience and a large amount of data mining. As shown in Figures 3(a) and 8(a), the XGBoost-LSTM model identified 2035 more bad samples than the XGBoost model (a total of 8,863 bad samples), and the number of false identifications of good samples decreased by 957. It means that bad customers are detected 23% more, and the false identification rate of samples predicted to be bad users decreased by 16% in the actual application of the model. The XGBoost-LSTM model thus

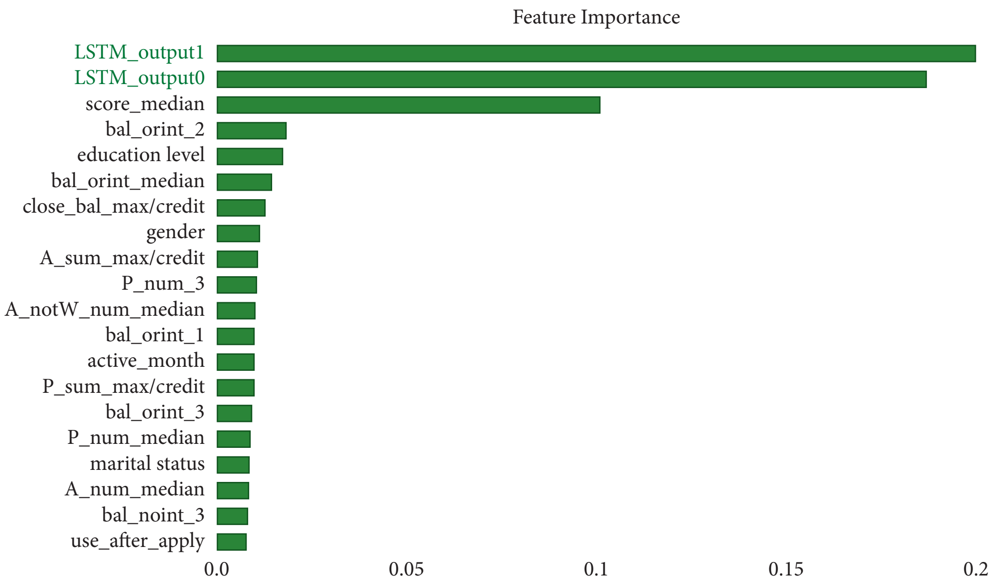

Figure 7: Feature importance index ranking of the XGBoost-LSTM model.

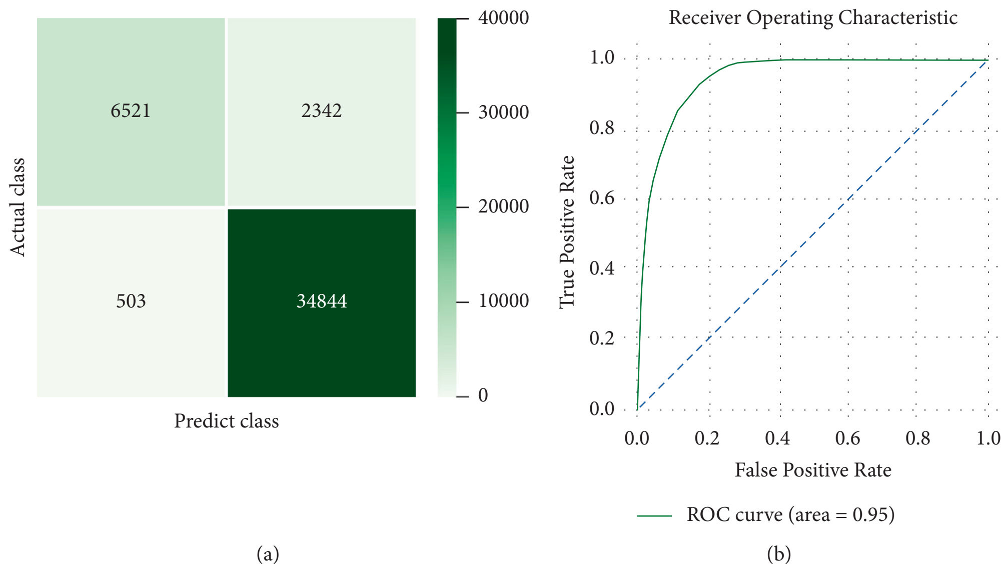

Figure 8: The confusion matrix and ROC of the XGBoost-LSTM model. (a) The confusion matrix of the test data. (b) The ROC curve of the test data.

Table 7: Data processing methods for modeling with different algorithms.

	XGBoost		XGBoost_LSTM	
	Data processing	Number	Data processing	Number
Basic information	—	19	—	19
Credit information	Monthly data Calculate statistics	6	Monthly data Calculate statistics	6
Billing information	Monthly data Calculate statistics	106	Monthly data Calculate statistics	106
Installment information	Monthly data Calculate statistics	113	Monthly data Calculate statistics	113
Transaction flow information	Feature extraction Calculate statistics	83	Interception or zero padding, splicing LSTM	2

Table 8: Model evaluation indicators for XGBoost model and XGBoost-LSTM model.

Algorithm	Train AUC	Train ACC	Test AUC	Test ACC	Test recall	Test precision
XGBoost	0.895	0.873	0.893	0.870	0.507	0.768
XGBoost-LSTM	0.954	0.937	0.953	0.936	0.736	0.928

obtains best prediction results on the data set used in this study. The results verify that the XGBoost-LSTM model can be realized without feature extraction and still have high classification accuracy.

6. Conclusions

In the method used in this study, features related to transaction flow had the highest importance weight, showing that the transaction flow data could effectively predict credit card default. Second, in the process of XGBoost modeling, the accuracy of default prediction mainly depended on feature extraction. It takes a lot of time to understand the specific meaning of each transaction type, but only when there is a deep understanding of the credit card business background can transaction types be correctly classified and useful features extracted. Third, when applying LSTM to process transaction flow data, it was only necessary to complement and splice the data, without any feature extraction work, which again confirms that the advantage of deep learning is that it does not require manual feature extraction. Finally, the XGBoost-LSTM fusion model combined basic, billing, and installment information, as well as PBOC branch and transaction flow, and it obtained extremely good test accuracy. This study shows that LSTM is an effective method for dealing with credit card transaction flow data.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

References

[1] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 34, no. 11, pp. 2767–2787, 2010.

[2] S. Bayraci and O. Susuz, "A deep neural network (DNN) based classification model in application to loan default prediction," *Theoretical and Applied Economics*, vol. 4, pp. 75–84, 2019.

[3] S. Fan, Y. Shen, and S. Peng, "Improved ML-based technique for credit card scoring in internet financial risk control," *Complexity*, vol. 2020, Article ID 8706285, 14 pages, 2020.

[4] Y. Guo, S. Jiang, F. Chen, Y. Li, and C. Luo, "Borrower-lender information fusion for P2P lending: a nonparametric approach," *Ingénierie des Systèmes d'Information*, vol. 24, no. 3, pp. 269–279, 2019.

[5] L. Zhu, D. Qiu, D. Ergu, C. Ying, and K. Liu, "A study on predicting loan default based on the random forest algorithm," *Procedia Computer Science*, vol. 162, pp. 503–513, 2019.

[6] H. Liu, "Credit risk prediction model based on machine learning algorithm," *Microcomputer Applications*, vol. 35, pp. 70–73, 2019.

[7] P. M. Addo, D. Guegan, and B. Hassani, "Credit risk analysis using machine and deep learning models," *Risks*, vol. 6, no. 2, 38 pages, 2018.

[8] M. Bakoben, T. Bellotti, and N. Adams, "Identification of credit risk based on cluster analysis of account behaviours," *Journal of the Operational Research Society*, vol. 71, no. 5, pp. 775–783, 2019.

[9] F. Butaru, Q. Chen, B. Clark, S. Das, A. W. Lo, and A. Siddique, "Risk and risk management in the credit card industry," *Journal of Banking & Finance*, vol. 72, pp. 218–239, 2016.

[10] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016.

[11] Y. Zhang and L. Chen, "A study on forecasting the default risk of bond based on XGBoost algorithm and over-sampling method," *Theoretical Economics Letters*, vol. 11, no. 2, pp. 258–267, 2021.

[12] X. Zhou, W. Zhang, and Y. Jiang, "Personal credit default prediction model based on convolution neural network," *Mathematical Problems in Engineering*, vol. 2020, Article ID 5608392, 10 pages, 2020.

[13] V. Bandi, D. Bhattacharyya, and D. Midhunchakkravarthy, "Prediction of brain stroke severity using machine learning," *Revue d'Intelligence Artificielle*, vol. 34, no. 6, pp. 753–761, 2020.

[14] T. Chou, M. Lo, and M. Lo, "Predicting credit card defaults with deep learning and other machine learning models," *International Journal of Computer Theory and Engineering*, vol. 10, no. 4, pp. 105–110, 2018.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] S. Wang, B. Yuan, and D. Wu, "A hybrid classifier for handwriting recognition on multi-domain financial bills based on DCNN and SVM," *Traitement du Signal*, vol. 37, no. 6, pp. 1103–1110, 2020.

[17] J. M. T. Wu, M. E. Wu, P. J. Hung, M. M. Hassan, and G. Fortino, "Convert index trading to option strategies via LSTM architecture," *Neural Computing and Applications*, pp. 1–18, 2020.

[18] J. Dong and X. Li, "An image classification algorithm of financial instruments based on convolutional neural network," *Traitement du Signal*, vol. 37, no. 6, pp. 1055–1060, 2020.

[19] I. E. Livieris, E. Pintelas, and P. Pintelas, "A CNN-LSTM model for gold price time-series forecasting," *Neural Computing and Applications*, vol. 32, no. 23, pp. 17351–17360, 2020.

[20] Y. Alghofaili, A. Albattah, and M. A. Rassam, "A financial fraud detection model based on LSTM deep learning technique," *Journal of Applied Security Research*, vol. 15, no. 4, pp. 498–516, 2020.

[21] Y. Qiu and J. Lu, “A visualization algorithm for medical big data based on deep learning,” *Measurement*, vol. 183, Article ID 109808, 2021.

[22] C. X. Ling, J. Huang, and H. Zhang, “AUC: a better measure than accuracy in comparing learning algorithms,” in *Proceedings of the 2003 Conference of the Canadian Society for Computational Studies of Intelligence*, vol. 2671, Halifax, Canada, 2003.