IPSJ SIG Technical Report
Vol.2019-ICS-194 No.11
2019/3/10

Application of XGBoost to Credit Scoring

Ce Chen[1,†1,a)] Soichiro Yokoyama[1] Tomohisa Yamashita[1] Hidenori Kawamura[1]

Abstract: With the continuous development of the credit industry, the competition and challenges faced by credit institutions, including various online lending platforms and banks, are becoming more and more serious. As the core competitiveness of an institution, credit evaluation is becoming more and more important. Credit scoring technology plays an important role in credit risk management. How to build a reliable credit score model to evaluate the credit risk of loan applicants has become an important research topic in academic and business circles. Traditional statistical methods and artificial intelligence methods are the two most commonly used models in the field of credit assessment. However, in practice, the performance of a single model is often not ideal. By integrating multiple base models, ensemble learning can effectively improve the prediction accuracy of the whole model.
For the credit scoring's imbalanced data and cost-sensitive, this paper improves XGBoost to make it reduce the cost. The AUC and cost-sensitive error rate are used as evaluation indexes to verify the validity of this model by comparing with other common models.

Keywords: XGBoost, focal loss, ensemble learning, credit scoring

1. Introduction

1.1 Overview of credit scoring

Credit assessment is the core process of credit management. It evaluates the credit status of customers based on their personal basic information and historical credit records, and then differentiates customers with low default risk from those with high default risk. Therefore, the credit score model is essentially a classification problem. The purpose of credit scoring is to classify the applicants into two types:applicants with good credit and applicants with bad credit. Applicants with good credit have great possibility to repay financial obligation. Applicants with bad credit have high possibility of defaulting. The basic idea of credit evaluation is to compare the characteristics of new customers with those of earlier customers in historical data. If the characteristics of the new customer are similar to those of the defaulting customer in history, the loan will be rejected;If the characteristics of a new customer are similar to those of a historically performing customer, the loan will be permitted . The accuracy of credit scoring is critical to financial institutions' profitability. Even 1% of improvement on the accuracy of credit scoring of applicants with bad credit will decreases a great loss for financial institutions[8].

The credit scoring model identifies financial variables that have statistical explanatory power in differentiating bad customers from good ones. The benefits obtained by developing a reliable credit scoring systems are[12]:

(1) Reducing the cost of credit analysis
(2) Enabling faster decision
(3) Insuring credit collections and diminish possible risk

1 Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Hokkaido, 060–0814, Japan
†1 Presently with Hokkaido University
a) chinsaku@complex.ist.hokudai.ac.jp

1.2 Characteristics of credit scoring

1.2.1 Imbalanced data

In the credit scoring data set, the number of positive and negative samples differs a lot. The data set of credit scoring is generally manually filtered beforehand, so the good credit sample is much more than the bad credit sample. So there will be an imbalance problem. When encountering imbalanced data, the traditional classification algorithm with the overall classification accuracy rate as the learning goal will pay too much attention to the majority class, thus reducing the classification performance of minority class samples. Most common machine learning algorithms do not work well for unbalanced data sets.

1.2.2 Cost Sensitive

Cost-sensitive learning is a method of providing different weights for different categories of samples, allowing machine learning models to learn. In a typical learning task, the weights of all samples are generally equal, but in some specific tasks, different weights can be set for the samples. In the problem of credit scoring, the loss caused by classifying a bad cred customer as a good credit customer is far greater than the loss of classifying a good credit customer to a bad credit customer. Therefore, in this case, it is necessary to avoid classifying bad credit customers as good credit customers.

1.3 Research objectives

The primary problem with credit scoring is to get the least cost. XGBoost is a very accurate model, but the processing power for cost sensitive is weak. So for this problem, the XGBoost model has been improved so that it can better solve the problem of cost sensitive.

2. Related works

2.1 Previous research

So far, the credit assessment field has mainly applied two basic methods: expert scoring model and credit scoring model based on statistics and machine learning. The expert scoring model was first used to solve the credit evaluation problem. Credit analysts make quantitative analyses of customer characteristics, such as solvency, loan amount, loan term, etc. , and then decide whether to grant the loan or not. However, this approach relies heavily on the subjective experience of experts and is inefficient. With the development of information technology, many scholars use statistical and machine learning methods to build credit scoring model. Common methods include linear discriminant analysis, logical regression, support vector machines, naive bayes, decision trees, k-Nearest Neighbor and neural network.

As for the performance of these models in the credit score, Iran[3] and Stefan[1] compared the common models in their paper, and the results showed that the performance of LR was better in the single model, while KNN and DT were not very ideal. Recently, there have been more and more studies on credit score problems using ensemble learning, and these papers all show that ensemble learning is more accurate than single model[1], [3], [6], [13], [17]. In data mining competitions such as Kaggle, competitors often use ensemble learning to integrate multiple models, thereby improving the performance of the overall model. Ensemble learning is a powerful tool to improve the accuracy of model prediction.

Ensemble learning, in simple terms, refers to the integration of multiple single learners to complete learning tasks. In order to get a good outcome, individual learners should be ”good but different”. That means, individual learners should have certain accuracy and there should be differences between each individual learner. Based on this point, this paper has showed three improved XGBoost model , to solve credit scoring problems.

Credit scoring is to solve the problem of cost sensitive in the data set of imbalanced data. There are two main ways to solve the problem of cost sensitive. One is resampling, and the other is to change the model structure.

Marque[10], Crone[5]’s paper uses the resampling to prove that oversampling is better than undersampling by changing the ratio of positive and negative samples. But oversampling tends to cause overfitting, and undersampling tends to cause underfitting. Garcia’s paper compares the various resampling methods and concludes that smote-based oversampling performances best. However, smote-based oversampling uses synthetic new data, which is not rigorous enough for credit scoring.

Bahnsen[2], Sahin[11]’s paper proposed a cost sensitive decision tree and a cost sensitive logistic regression. But decision tree and logistic regression are single models, and the accuracy is not as good as ensemble model.

In ensemble model, XGBoost has a good performance with high accuracy. XGBoost uses cart as a base model. Compared with neural networks, it has a clear structure and good interpretability. For credit scoring problem, there are three papers using XGBoost as the research model. Zhou[18]’s paper uses accuracy as the evaluation standard, compares XGBoost with svm, lr and other single models, and finds that XGBoost performs best. Xia’s paper[15] compared the XGBoost tuning method and did not change the structure of XGBoost. Liu’s paper[16] proposes a cost sensitive XGBoost(CSXGBoost) to better solve cost sensitive problems.

This paper presents improved XGBoost by changing the structure of cost function and evalution metrics.The model will be compared with CSXGBoost to prove that the third model has a better performance.

2.2 Public data set

Table 1 Public dataset

Datasets	Samples	Features	Good/bad
German	1000	24	700/300
Australian	690	14	307/383
Taiwan	30000	23	23364/6636
Japanese	690	15	307/383
Qianhai	40000	491	34740/5260

In the credit scoring paper, a lot of data sets are used. However, most of the papers only introduce the parameters of the data set, and do not give the source of the data. There are currently 5 public data sets found. German dataset, Australian datasets, Taiwan datasets, Japanese datasets, Qianhai datasets.

3. XGBoost

XGBoost’s full name is eXtreme Gradient Boosting, which is a c++ implementation of Gradient

Boosting Machine(GBM)[7], presented in 2016 by Ph. D. student Chen Tianqi from the University of Washington[4]. XGBoost has performed well in all kinds of data mining competitions, including 2017 ACM RecSys challenge (first place), KDD Cup 2016 competition (first place), Caterpillar Tube Pricing competition (Kaggle, first place), Airbnb New User Bookings (Kaggle, second place) etc.

3.1 The basic principle of XGBoost

XGBoost is an efficient implementation of the GB algorithm. The base learner in XGBoost can be either a CART (gbtree) or a linear classifier (gblinear).

XGBoost adds a regularization term to the objective function. When the base learning is CART, the regularization term is related to the number of leaf nodes of the tree and the value of the leaf node.

$$L(\varphi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{1}$$

$$where\ \Omega(f) = \gamma T + \frac{1}{2}\lambda \parallel w \parallel \tag{2}$$

$L(\varphi)$ is objective that need to be optimized. $\Omega(f)$is the regularization that prevent overfitting.The GB uses the Loss Function to calculate the pseudo-residue for the first derivative of f(x) for learning to generate fm(x). XGBoost uses not only the first derivative but also the second derivative.

The tth loss:

$$L^{(t)} = \sum_{i=1} l(y_i, \hat{y}^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (3)$$

Do a second-order Taylor expansion on the above formula: g is the first derivative and h is the second derivative.

$$L^{(t)} \cong \sum_{i=1}^{n} \left[l(y_i, \hat{y}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \qquad (4)$$

$$where\ g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)} \qquad (5)$$

$$and\ h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}) \qquad (6)$$

The above metric for finding the best segmentation point in the CART regression tree is to minimize the mean square error. The criterion for XGBoost to find the segmentation point is to maximize, lamda, gama is related to the regularization term.

$$L_{split} = 1/2[\frac{(\sum_{i \in I_L} g_i)^2}{(\sum_{i \in I_L} (h_i + \lambda))} + \frac{(\sum_{i \in I_R} g_i)^2}{(\sum_{i \in I_R} (h_i + \lambda))} - \frac{(\sum_{i \in I} g_i)^2}{(\sum_{i \in I} (h_i + \lambda))}] - \gamma \qquad (7)$$

L_{split} is the reduction value before and after splitting.

In addition to the regular items mentioned above, XGBoost uses two additional methods to prevent overfitting. The first method is the Shrinkage proposed by Friedman. The tree lifting algorithm adds a tree structure to the existing model in each loop, and the reduction technique is similar to the learning rate, which will add new trees. The weight is multiplied by a scaling factor η to reduce the impact of a single tree and provide optimization space for the latter tree.

The second method is column subsampling, which is often used in random forests. In general, column sampling prevents overfitting better than row sampling and speeds up parallel algorithms.

3.2 XGBoost basic features

XGBoost's "extreme" doesn't mean it's extremely accurate. In fact, XGBoost's accuracy is roughly the same as Sci-Kit Learn (another implementation of GBM). “ extreme ” actually means that XGBoost runs extremely fast, which is determined by the following three features of XGBoost:

(1) Sparse perception

The gradient lifting tree algorithm is particularly suitable for categorical features. In practical problems, most of the category-type features are 0, which is more common. XGBoost can automatically identify the location of non-zero data, improving the efficiency of the algorithm.

(2) Parallel

The most time consuming step of the gradient lifting tree algorithm is to order continuous features. XGBoost can independently sort each column (feature) so that the sorting work can be divided into several parallel threads for the CPU to complete.

(3) Approximate division

In order to find the most efficient partitioning of continuous features, the gradient lifting tree needs to put all the data into memory and sort it at the same time, which greatly limits the size of the data set. XGBoost proposes a parallel approximation histogram algorithm for efficient generation of candidate segmentation points. As long as the histogram group spacing is appropriate, XGBoost can get the same partitioning effect in less time.

3.2.1 XGBoost for credit scoring

Compared to the black box of the neural network, the training process of XGBoost is easier to understand. XGBoost is an improved algorithm based on GBDT, which can give scores of each feature during the training process, thus indicating the importance of each feature to model training. Feature importance is calculated by sorting and sorting each feature in the dataset. The feature score can be viewed as the number of times the decision tree is used to separate. By outputting the importance of features, we can have a clear understanding of each feature. In the case of manual review, we can judge the credit of the sample better.

For the credit scoring's imbalanced data and cost sensitive, the improvement for XGBoost is that it can better handle the problem of credit scoring and reduce the cost. At the same time, the improved XGBoost also inherits the advantages of XGBoost, with high accuracy and clear training process. The improved model can better meet the needs of creditor and further reduce losses.

4. Evaluation Criterion

4.1 Area under Curve

In the binary classification problem, the predicted results of many machine learning models are probabilities. When calculating accuracy, it is necessary to set a threshold, and then convert the probability into a category. The setting of threshold greatly affects the performance of accuracy. In data mining competitions such as Kaggle, AUC is often used as a model evaluation indicator, because it can avoid transforming predictive probabilities into categories.

The AUC is called Area under Curve, which means the Area under the Receiver operating characteristic Curve. In order to explain the ROC curve, we introduce Confusion Matrix at first:

Table 2 Confusion Matrix

		Predicted Value	
		0	1
True Value	0	True Negative	False Positive
	1	False Negative	True Positive

(1) If an instance is a positive class and is predicted to be a positive class, that is True Postive TP

(2) If an instance is positive but predicted to be Negative, it is False Negative FN.

(3) If an instance is negative but is predicted to be a positive class, that is False Postive FP

(4) If an instance is Negative class, but is predicted to be Negative class, that is True Negative TN.

In the process of calculating ROC curve, the prediction probability is first sorted from high to low, and each probability is taken as a threshold, thus generating multiple confusion matrices. For each confusion matrix,

the true positive rate = true positive/(true positive + false negative)

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8}$$

false positive rate = false positive/(false positive + false negative)

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{9}$$

ROC curve can be calculated by taking the false positive rate as X-axis and the true positive rate as Y-axis.

But in many cases, the ROC curves of different models (classifiers) intersect with each other, and it is not clear which model works better. As a value, AUC is more convenient to judge the pros and cons of the model. The larger AUC value is, the better the model classification effect is. Therefore, AUC was used as the evaluation index of the model in the experiment.

4.2 Cost Sensitive

Cost sensitive refers to a method that provides different weights to different types of samples. In normal learning tasks, the weight of all samples is generally equal, but in some specific tasks, different weights can be set for the samples. For example, in medical treatment, "the cost of misdiagnosing a patient as a healthy person" is different from "the cost of misdiagnosing a healthy person as a patient". In the case of credit scoring, the cost of splitting bad customers into good ones is greater than the cost of splitting good ones into bad ones.

Table 3 Cost Sensitive

		Predicted Value	
		good	bad
True Value	good	0	Cost(1)
	bad	Cost(N)	0

Cost(1) indicates the cost of misjudging a good customer to a bad customer. Cost(N) indicates the cost of misjudged a bad customer as a good customer. In the credit scoring problem, there is obviously that $Cost(N) > Cost(1)$.

There are many ways to measure cost sensitive. For the credit scoring problem, since the bank and other creditors want to know the final loss value, the overall loss value is used to measure the pros and cons of the model.

There are three main types of cost sensitive formulas that use the overall loss value as a measure. L1 and L2 are the amount of FP, FN, C_{FP} and C_{FN} are the loss of FP, FN.

Table 4 Parameter explanation

	Amount	Loss
FP	L1	C(FP)
FN	L2	C(FN)

$$\frac{1}{C_{FN} + C_{FP}}(C_{FP}L_1 + C_{FN}L_2) \tag{10}$$

$$C_FPL_1 + C_FNL_2 \tag{11}$$

$$EMC = 1/Total(C_FPL_1 + C_FNL_2) \tag{12}$$

4.3 Comparison of cost sensitive formula

$$\frac{1}{C_{FN} + C_{FP}}(C_{FP}L_1 + C_{FN}L_2) \tag{13}$$

Table 5 Cost output by formula (13)

Model	German	Australian
LR	29. 6	8. 8
DT	30.5	14. 2
RF	31. 8	9. 1
XGBoost	27. 5	8. 6

$$C_{FP}L_1 + C_{FN}L_2 \tag{14}$$

Table 6 Cost output by formula (14)

Model	German	Australian
LR	177. 8	51
DT	191. 6	86
RF	190.8	52. 6
XGBoost	165. 2	43

$$EMC = \frac{1}{Total}(C_{FP}L_1 + C_{FN}L_2) \tag{15}$$

All three formulas can be used as criteria for judging the pros and cons of the model. However, in order to facilitate the comparison of various data sets, EMC[14] is adopted as the formula of cost sensitive. Parameter Total is the total number of samples. The cost of FP and FN is calculated differently in different papers. Some papers use the features in the sample in the calculation process and cannot be used with all data sets. In this paper, $C_{FN} = N = 5, C_{FP} = 1$.

Table 7 Cost output by EMC (formula(15))

Model	German	Australian
LR	0.875	0.393
DT	0.975	0.620
RF	0.954	0.394
XGBoost	0.81	0.381

$$m = G \cup B \tag{16}$$

$$E_{cost} = \frac{1}{m}(\sum_{x \in G}(f_x \neq y) \times Cost(FP) + \sum_{x \in B}(f_x \neq y) \times Cost(FN)) \tag{17}$$

G is the data set for good customers and B is the data set for bad customers.

$$E_{cost} = \frac{1}{m}(\sum_{x \in G}(f_x \neq y) + \sum_{x \in B}(f_x \neq y) \times n) \tag{18}$$

4.4 k-fold cross Validation

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model. It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

The general procedure is as follows:

(1) Shuffle the dataset randomly.
(2) Split the dataset into k groups
(3) For each unique group:
  - Take the group as a hold out or test data set
  - Take the remaining groups as a training data set
  - Fit a model on the training set and evaluate it on the test set
  - Retain the evaluation score and discard the model
(4) Summarize the skill of the model using the sample of model evaluation scores

Importantly, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times. In this study, 5-fold cross Validation was used to improve accuracy, and the result was the mean of five results.

5. Datasets

This paper uses 2 data sets: German datasets, Taiwan datasets.

German credit datasets come from uci database(http://archive. ics. uci. edu/ml/datasets. html). The data in the data numeric file, which is widely used in the verification of the credit scoring model. It contains 1, 000 loan application records, of which 700 are trusted " good " customers and 300 are " bad " customers who default. The original data is described by 19 attributes. The officially digitized file uses onehot coding to convert the nominal attribute into a dummy variable. Finally, the loan application record is described by 24 attributes.

Taiwan credit datasets come from uci database(http://archive. ics. uci. edu/ml/datasets. html). The data set contains a total of 30000 samples, of which 23364 are good credit customers and 6636 are credit bad customers. For each samples, it contains 23 features.

6. XGBoost with focal loss

The model uses focal loss[9] as the cost function in XGBoost and analyzes its performance while customizing eval_metric. In this study, 5-fold cross Validation was used to improve accuracy, and the result was the mean of five results.

6.1 Evaluation metrics

To get the lowest cost, the model itself needs to be optimized. XGBoost can customize the evaluation metrics. Objective and eval_metric are the two parameters of XGBoost. Objective participates in training, and the goal of training is to lower objective.

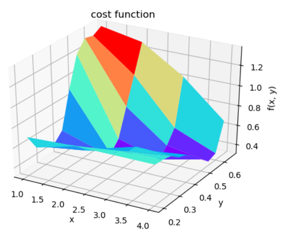


Fig. 1 Cost graphics in German dataset

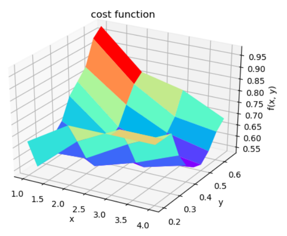


Fig. 2 Cost graphics in Taiwan dataset

Evalmetric is the metric needed to verify the data, and evalmetric is not involved in the training. Although evalmetric does not participate in the training, it is an evaluation metric. When the evalmetric reaches the optimal value, XGBoost stops the training.

6.1.1 Weight

XGBoost allows for instance weighting during the construction of the DMatrix, as you noted. This weight is directly tied the instance and travels with it throughout the entire training. Thus it is included in the calculations of the gradients and hessians, and directly impacts the split points and traing of an XGBoost model.

6.1.2 Threshold

XGBoost outputs probability values, marked as prob_y. The default threshold is 0.5. If prob_y is greater than threshold, the predicted value of $y(predict\_y) = 1$, otherwise, the predicted value of $y(predict\_y) = 0$. Make three-dimensional graphics based on these two variables. X is weight of minority class,y is threshold,z is the formulated cost.

Figure 1 is the outcome of German dataset.Figure 2 is the outcome of Taiwan dataset.It can be seen from the graph that by using the customized Eval_metric and using threshold and weight as parameters, the cost of the model can be reduced.

6.2 Objective

Focal loss is a cost function used for object detection. Currently, target detection frameworks are generally divided into two types: two-stage detection framework based on candidate regions

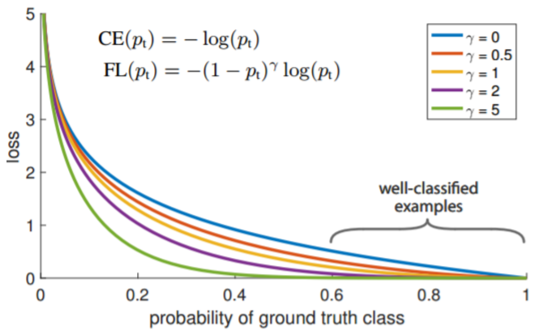


Fig. 3 Proportion of well classified samples in loss

(such as fast r-cnn series), and one-stage detection framework based on regression (such as yolo, SSD). Two-stage has a good effect, while one-stage has a fast but poor effect. The author hopes to figure out why the accuracy of the one-stage detector is not high. The author gives the explanation that the former positive and negative samples are not balanced.

6.2.1 Problems caused by imbalanced samples

(1) training is inefficient as most locations are easy negatives that contribute no useful learning signal;

(2) The easy negatives can overwhelm training and lead to degenerate models.

Since most of them are simple and easy to divide negative samples (samples belonging to the background), the training process can not fully learn the information belonging to those category samples; secondly, there are too many negative samples that are easy to divide, which may cover other class samples. The role of these simple and easily divided negative samples still produce a certain degree of loss, see the blue curve below, the number will play a major role in the loss, so it dominates the direction of the gradient update, hiding important information). Figure 3 come from the paper Focal Loss for Dense Object Detection[9].

6.2.2 Solution: Focal Loss

The authors propose a new loss function, this function can reduce the weight of the easily classified samples, so that the model can focus more on the samples that are difficult to be classified.

Focal loss is a modification based on the cross entropy loss function.

y' is the output of the activation function, so it's between 0 and 1. It can be seen that for positive samples, the larger the output probability is, the smaller the loss will be. For a negative sample, the smaller the output probability, the smaller the loss. At this time, the loss function is relatively slow in the iteration process of a large number of simple samples and may not be optimized to the optimal level.

Focal loss has improved this shortcoming. First, a factor is added to the original one, where $gamma > 0$ reduces the loss of easily classified samples. It makes them more focused on the difficult, mislaid samples.

For example, if γ is 2, for positive samples, the predicted result of 0.95 must be a simple sample, so (1-0.95) γ is small, and the loss function becomes smaller. A sample with a prediction probability of 0.3 has a relatively large loss. For negative samples, the prediction of 0.1 should be much smaller than the prediction of 0.7. For a prediction probability of 0.5, the loss is reduced by only 0.25 times, so more attention is paid to such indistinguishable samples. In this way, the influence of simple samples is reduced, and the effect of a large number of samples with small prediction probability may be more effective when combined.

I think there is a certain similarity between credit scoring and object detection. Most of the credit scoring sample sets are good credit samples, and the bad credit samples are only a small part. Most samples of good credit are very different from samples of bad credit. Only a small part is likely to have similar sample characteristics to bad credit. Therefore, it is necessary to reduce the weight of well-classified samples and increase the weight of hard-to-classify samples

6.3 Model structure

The cost function used in this paper is formula :

$$FL = -\sum_{i=1}^{m} \alpha y(1-p(x_i))^{\gamma} log(p(x_i)) + (1-y)p(x_i)^{\gamma} log(1-p(x_i)) \tag{19}$$

To apply focal loss to XGBoost, the first and second derivatives of focal loss should be required.

After calculation, the first derivative is:

$$\begin{aligned} grad =& \alpha y(1-p(x_i))^{\gamma}(1-p(x_i)-P(x_i)\gamma log(p(x_i))) \\ & -(1-y)p(x_i)^{\gamma}(p(x_i)-(1-p(x_i))\gamma log(1-p(x_i))) \end{aligned} \tag{20}$$

The second derivative is:

$$\begin{aligned} Hess =& \alpha y p(x_i)(1-p(x_i))^{\gamma} \\ & [\gamma log(p(x_i))(p(x_i)\gamma + p(x_i) - 1) + 2\gamma + 1)p(x_i) - 2\gamma - 1] + \\ & (1-y)\{p(x_i)^{\gamma}(1-p(x_i))[\gamma log(1-p(x_i))(1-p(x_i)) \\ & (\gamma - \gamma p(x_i) - p(x_i)) + (2\gamma+1)(1-p(x_i)) - 2\gamma - 1]\} \end{aligned} \tag{21}$$

In order to verify the correctness of the results, focal loss was compared with logloss.

Logloss:

$$logloss(y_i, x_i) = -\frac{1}{N}\sum_{i=1}^{N}\{(y_i)log[p(x_i)] + (1-y_i)log[1-p(x_i)]\} \tag{22}$$

When $\alpha = 1, \gamma = 0$,

$$FL(\alpha = 1, \gamma = 0) = logloss \tag{23}$$

After calculation, it is proved that the first derivative and the second derivative are calculated correctly.

There are four custom parameters in the model, weight, threshold, α and γ. In the process of reading the source code, it was found that weight and alpha had the same function.

so weight was removed and only three parameters of threshold, α and γ were used.

6.4 Outcome

This model uses two data sets, German datasets and Taiwan datasets. In the model, there are three factors that affect cost, threshold, α and γ.

For the three parameters threshold, α and γ, the effects of the remaining parameters on cost were observed when two parameters were fixed. As can be seen from the table, threshold, alpha, gamma, have a significant impact on the cost. When adjusting the parameter value, we can get lower cost than XGBoost.

6.4.1 German dataset

Table 8 9 10 uses the German dataset.

Table 8: α=1, γ=0.

Table 9: α=1, $threshold$=0,

Table 10: γ=0, threshold=0.5

Table 8 The change of cost is by threshold

threshold	cost
0. 08	0.558
0. 09	0.542
0.1	0.539
0.2	0.545
0.3	0.614
0.4	0.693

Table 9 The change of cost is by γ

γ	cost
0	0.813
0.5	0.797
1	0.800
2	0.822
3	0.817
4	0.832

Best cost=0.551, α=5, threshold=0.5, γ=1

6.4.2 Taiwan dataset

Table 11 12 13 uses the Taiwan dataset.

Table 11: α=1, γ=0.

Table 12: α=1, threshold=0,

Table 13: γ=0, threshold=0.5

Best cost=0.550, α=4, threshold=0.5, γ=2

6.5 Model Comparison

In order to verify the improvement of the model, we use CSXGBoost to compare with the new model. The paper uses ARR(Annualized Rate of Return) and AUC as the criteria for judging the performance of the model. It can be seen from the figure4 that CSXGBoost performs better than other models.

Annualized Rate of Return:

$$ARR = \left(\frac{P+G}{P}\right)^{\frac{1}{N}} - 1 \tag{24}$$

Where P is the principle, G is the interest gain, N is the number

Table 10 The change of cost is by α

α	cost
1	0.813
2	0.671
3	0.636
4	0.582
5	0.567
6	0.575

Table 11 The change of cost is by threshold

threshold	cost
0. 08	0.665
0. 09	0.636
0.1	0.612
0.2	0.552
0.3	0.616
0.4	0.686

Table 12 The change of cost is by γ

γ	cost
0	0.742
0.5	0.741
1	0.77
2	0.741
3	0.745
4	0.746

Table 13 The change of cost is by α

α	cost
1	0.742
2	0.641
3	0.581
4	0.553
5	0.554
6	0.557

of years

paper is used as the criterion.

$$EMC = \frac{1}{Total}(C_{FP}L_1 + C_{FN}L_2) \tag{25}$$

CSXGBoost also uses logloss as the cost function. It modifies the form of sigmoid.

$$logloss(y_i, x_i) = -\frac{1}{N}\sum_{i=1}^{N}\{(y_i)log[p(x_i)] + (1-y_i)log[1-p(x_i)]\} \tag{26}$$

The original:

$$p(x_i) = \frac{1}{1+e^{-f(x_i)}} \tag{27}$$

CSXGBoost:

$$p(x_i) = \frac{1}{1+e^{-2\sigma f(x_i)-2\eta}} \tag{28}$$

Where,

$$\delta = \frac{C_{FP}+C_{FN}}{2}, \eta = \frac{1}{2}log\frac{C_{FP}}{C_{FN}} \tag{29}$$

$g_i = 2\delta[y - p(x_i)]$ is the first derivative of the cost function of CSXGBoost, $h_i = 4\delta^2 p(x_i)[1-p(x_i)]$ is second derivative of cost function. German datasets Taiwan datasets

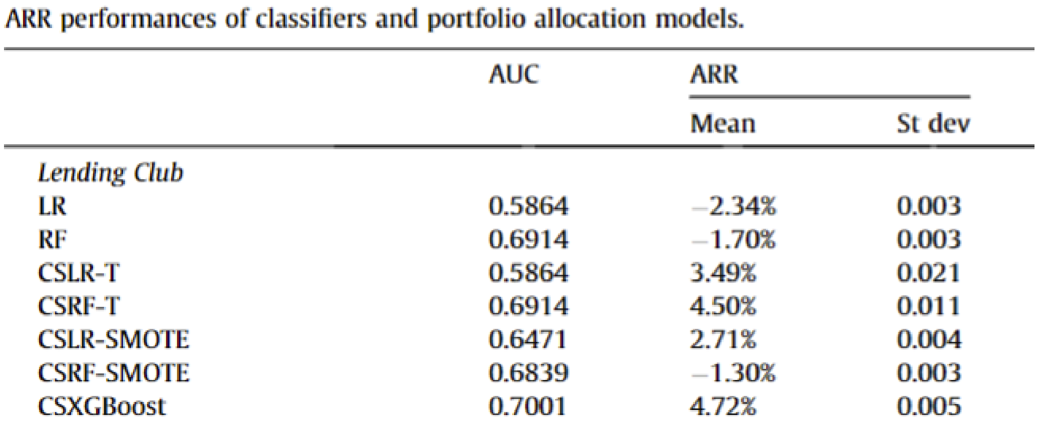
Table 14 Model comparison in German dataset

Model	Const(n=5)
XGBoost	0.813
CSXGBoost	0.801
XGBoost_focal	0.551

Table 15 Model comparison in Taiwan dataset

Model	Const(n=5)
XGBoost	0.742
CSXGBoost	0.730
XGBoost_focal	0.550

The data set uses the german dataset and the taiwan dataset. It can be seen from the figure that XGBoost_focal performs better(



ARR performances of classifiers and portfolio allocation models.

	AUC	ARR	
		Mean	St dev
Lending Club			
LR	0.5864	−2.34%	0.003
RF	0.6914	−1.70%	0.003
CSLR-T	0.5864	3.49%	0.021
CSRF-T	0.6914	4.50%	0.011
CSLR-SMOTE	0.6471	2.71%	0.004
CSRF-SMOTE	0.6839	−1.30%	0.003
CSXGBoost	0.7001	4.72%	0.005

Fig. 4 Model comparison

lower cost).

7. Conclusion

With the continuous progress and development of modern life, more and more transactions are subject to credit risks, and debtors' credit needs to be specifically considered. Therefore, a reliable credit score model is very necessary.

In the issue of credit scoring, the positive and negative samples of the data set are unbalanced. Solving cost sensitive in imbalanced data is the problem of credit scoring. Therefore, the model proposed in this paper try to solve the cost sensitive problem of credit scoring. It can be seen from the result analysis that the model proposed in this paper has a good effect on reducing cost.

XGBoost is a very good model. After this improvement, it can handle the problem of credit scoring's imbalanced data and cost sensitive better, reducing the cost, thus reducing the losses of banks and other creditors.

References

[1] Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J. and Vanthienen, J.: Benchmarking state-of-the-art classification algorithms for credit scoring, *Journal of the operational research society*, Vol. 54, No. 6, pp. 627–635 (2003).

[2] Bahnsen, A. C., Aouada, D. and Ottersten, B.: Example-dependent cost-sensitive logistic regression for credit scoring, *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, IEEE, pp. 263–269 (2014).

[3] Brown, I. and Mues, C.: An experimental comparison of classification algorithms for imbalanced credit scoring data sets, *Expert Systems with Applications*, Vol. 39, No. 3, pp. 3446–3453 (2012).

[4] Chen, T. and Guestrin, C.: Xgboost: A scalable tree boosting system, *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, ACM, pp. 785–794 (2016).

[5] Crone, S. F. and Finlay, S.: Instance sampling in credit scoring: An empirical study of sample size and balancing, *International Journal of Forecasting*, Vol. 28, No. 1, pp. 224–238 (2012).

[6] Devi, C. D. and Chezian, R. M.: A relative evaluation of the performance of ensemble learning in credit scoring, *Advances in Computer Applications (ICACA), IEEE International Conference on*, IEEE, pp. 161–165 (2016).

[7] Friedman, J. H.: Greedy function approximation: a gradient boosting machine, *Annals of statistics*, pp. 1189–1232 (2001).

[8] Hand, D. J. and Henley, W. E.: Statistical classification methods in consumer credit scoring: a review, *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, Vol. 160, No. 3, pp. 523–541 (1997).

[9] Lin, T.-Y., Goyal, P., Girshick, R., He, K. and Dollár, P.: Focal loss for dense object detection, *IEEE transactions on pattern analysis and machine intelligence* (2018).

[10] Marqués, A. I., García, V. and Sánchez, J. S.: On the suitability of resampling techniques for the class imbalance problem in credit scoring, *Journal of the Operational Research Society*, Vol. 64, No. 7, pp. 1060–1070 (2013).

[11] Sahin, Y., Bulkan, S. and Duman, E.: A cost-sensitive decision tree approach for fraud detection, *Expert Systems with Applications*, Vol. 40, No. 15, pp. 5916–5923 (2013).

[12] Tsai, C.-F. and Wu, J.-W.: Using neural network ensembles for bankruptcy prediction and credit scoring, *Expert systems with applications*, Vol. 34, No. 4, pp. 2639–2649 (2008).

[13] Wang, G., Hao, J., Ma, J. and Jiang, H.: A comparative assessment of ensemble learning for credit scoring, *Expert systems with applications*, Vol. 38, No. 1, pp. 223–230 (2011).

[14] West, D.: Neural network credit scoring models, *Computers & Operations Research*, Vol. 27, No. 11-12, pp. 1131–1152 (2000).

[15] Xia, Y., Liu, C., Li, Y. and Liu, N.: A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring, *Expert Systems with Applications*, Vol. 78, pp. 225–241 (2017).

[16] Xia, Y., Liu, C. and Liu, N.: Cost-sensitive boosted tree for loan evaluation in peer-to-peer lending, *Electronic Commerce Research and Applications*, Vol. 24, pp. 30–49 (2017).

[17] Yao, P.: Credit scoring using ensemble machine learning, *Hybrid Intelligent Systems, 2009. HIS'09. Ninth International Conference on*, Vol. 3, IEEE, pp. 244–246 (2009).

[18] Zhou, X., Cheng, S., Zhu, M., Guo, C., Zhou, S., Xu, P., Xue, Z. and Zhang, W.: A state of the art survey of data mining-based fraud detection and credit scoring, *MATEC Web of Conferences*, Vol. 189, EDP Sciences, p. 03002 (2018).