Anomaly credit data detection based on enhanced Isolation Forest

Xiaodong Zhang1 · Yuan Yao1 · Congdong Lv1 · Tao Wang2

Received: 22 February 2022 / Accepted: 20 April 2022 / Published online: 26 April 2022
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

Abstract
In view of the real-world problem of falsity and errors credit data, and the performance degradation of the credit evaluation model caused by these problems, we proposed an outlier detection algorithm, which considered two characteristics of class-imbalance and cost-sensitive in credit data. We use an anomaly detection model called EIF to optimize the credit evaluation models. EIF uses the EasyEnsemble algorithm to construct balanced data sets, and train an Isolation Forest model for anomaly detection by the balanced datasets with different disturbances. On the one hand, the balanced dataset ensures that the class-imbalance problem is solved by undersampling, on the other hand, each sub-model learns from the overall minority class samples in order to solve the cost-sensitive problem. Experiments were performed on UCI German dataset, and the test set with fake data was constructed by correlation. Compared with other anomaly detection algorithms in common credit evaluation models, the EIF-optimized model has a higher F1 score and a lower cost-sensitive error rate. In conclusion, the EIF model is effective in enhancing the performance of the credit evaluation model for forged credit datasets.

Keywords Credit evaluation · Anomaly detection · Class-imbalance · Cost-sensitive · EasyEnsemble · Isolation forest

1 Introduction

The rapid economic development has promoted change of the public’s consumption concept. New forms of consumption such as prepaid and credit consumption have also promoted the development of the credit industry rapidly. The core of the credit business is to use the credit evaluation technology to reliably evaluate the customer’s credit, and control the risks generated in the business. In the process of credit evaluation, although various institutions can accurately evaluate the customer’s credit according to the collected data, the credit evaluation still based on a large amount of credible data. If abnormal data occurs due to forgeries and errors, credit evaluation will definitely be greatly affected. Therefore, the detection and processing of abnormal data has become the focus of credit evaluation technology.

The principle of credit evaluation is establishing a credit evaluation model based on various information data of the individual being evaluated, to find the individual characteristics of credit. The credit evaluation of future customers is based on the previous model. The initial credit evaluation work is largely based on the practitioners’ own experience to evaluate the customer’s credit, which is costly and is not conducive to mass use [1]. With the development of big data technology, machine learning methods are more and more used in credit evaluation work. But because of the seriousness of credit evaluation work, strongly interpretability models are used in most works. Vojtek et al. [2] use linear discriminant analysis (LDA) and logistic regression (LR) models for credit evaluation. The author explains that these two models are widely used in bank credit evaluation work due to their simplicity and strong interpretability. Uddin et al. [3] uses random forest (RF) model to assess the credit risk of micro-enterprises, and conducts multi-dimensional analysis on missing data. Such models include decision tree (DT) [4, 5], naive Bayes (NB) [6, 7], support vector machine (SVM) [8–10] et al. Neural networks have achieved good results in credit evaluation also. But because of the poor interpretability, they are not widely used in practice [9–11].

Anomaly detection also known as outlier detection in some literature. Anomaly detection assumes that a certain feature of the anomaly(outlier) is different from the
The distance-based outlier detection algorithm assumes that the normal are densely distributed and the anomalies are sparsely distributed. The distance between the sample and its neighbors is used as judgment index to detect abnormal samples [11]. In the research of Ren and Liu [12], the k-nearest-neighbors (KNN) was used to perform anomaly detection pre-processing on network intrusion behaviors, and obtained high-quality dataset was used for traditional model training. The density-based algorithm is a variant of the distance-based algorithm. It calculates the density difference between the sample and its neighbors to determine whether it is an anomaly [13, 14]. Campos [15] comprehensively compares the performance of representative outlier detection methods based on distance and density, and the Local Outlier Factor (LOF) algorithm, which is based on local density, has the best performance among such algorithms. The classification-based algorithm uses the traditional classification model as anomaly detection, trains a single-classification model on normal data, and judges the outliers that do not belong to the normal class [16]. The Isolation Forest (iForest) [17] randomly selects attributes and values to recursively divide the dataset to construct a tree structure. Anomalies are more likely to be isolated near the root of the tree, while normal points are deeper in the isolation tree. iForest has linear time complexity and is not affected by the sample dimension, so it performs well in medium and high dimensions [18].

There are two problems in the actual use of credit data: On the one hand, because the credit industry will conduct manual screening in the business process, samples with obvious low credit evaluation will be rejected, resulting in relatively few bad samples and insufficient description of the characteristics of bad credit. As a result, causing class-imbalance problem. On the other hand, the loss caused by misjudging bad credit samples as good credit samples is much larger than that of judging good credit samples as bad credit samples, resulting in the cost-sensitive problem. The class-imbalance problem is usually solved by sampling methods: the undersampling method randomly discards the majority of samples, which may cause the loss of important features; the oversampling method replicates the sampling of the minority class samples for many times, which will lead to overfitting. The SMOTE [19] algorithm uses the idea of oversampling to interpolate between the minority class samples in the data, which realizes the expansion of the samples and reduces the degree of overfitting. Liu et al. [20] proposed two methods, EasyEnsemble and Balance-Cascade, for the idea of ensemble learning, which combines the advantages of bagging and boosting ensemble learning methods, and uses multiple undersampled base models to form an overall missing ensemble model. This study overcomes the defect that simple undersampling loses most of the features. Cost-sensitive problems are usually solved by cost-sensitive learning. Frumosu et al. [21] explored the relationship between cost and quality in industry using cost-sensitive learning, mentioning the importance of cost metrics. But the cost of misjudgment is difficult to measure in credit evaluation work. Therefore, in this paper, each sub-model is allowed to learn from all few classes of samples to make the model as a whole more focused on bad credit samples.

In this paper, we use the correlation between attributes and credit to generate anomalies that can deceive the credit evaluation model, and use these anomalies to simulate fake samples in real work. After that, we use the idea of Easy-Ensemble to construct a balanced training data set, and through sample perturbation and attribute perturbation, train a better iForest model called EIF model. The EIF model can detect the fake data which generated in the previous article and improve the performance of the traditional credit evaluation model.

2 Existing work

In building the credit anomaly detection model, we used some ideas of the existing work. To help the reader understand the credit anomaly detection model in this study, this section introduces the core ideas of algorithms, Isolation Forest(iForest) and EasyEnsemble, used in the model.

2.1 Isolation forest

Isolation Forest(iForest) is an anomaly detection algorithm proposed by Liu et al. in 2012 [17]. The nodes of the isolation tree are defined $T$. $T$ is divided into external nodes and internal nodes: nodes without child nodes are called external nodes, and the remaining nodes are called internal nodes. The internal node consists of attribute $q$, the split value $p$ of attribute $q$, and two child nodes ($T_p$, $T_q$). $p$ is a random value between the maximum and minimum values of attribute $q$. In an isolation tree, $q$ and $p$ are used as the separation conditions of nodes. Each node compares the $q$ value of the new sample with $p$, and judges that the new sample belongs to $T_p$ or $T_q$.

When training an isolation tree on a $d$-dimensional dataset $X = \{x_1, x_2, \ldots, x_n\}$, $X' \subset X$ is obtained by sampling $X$ with size $\psi$. The isolation tree randomly selects the attribute $q$ and the split value $p$ to recursively split $X'$ until each node contains only one sample or all samples within a node have the same value. The dataset is sampled $n$ times to train different isolation trees, and all the different isolation trees constitute the isolation forest together.

Define the path length $h(x)$ of sample $x$ as the number of edges in the path from the root node of the isolation tree to the external node containing $x$. In the assumption, anomalies usually have short path lengths in the tree. $E(h(x))$ is the
average path length of x in the forest, and the calculation method of the anomaly score of x is as follows:

\[ s(x, \psi) = 2^{\frac{\ln(n)}{n^\psi}} \]  

(1)

\( H(i) \) is the harmonic number, which is usually estimated by the sum of \( \ln(i) \) and Euler’s constant (0.5772156649); \( c(\psi) \) is the average path length from the isolation tree with sampling number \( \psi \) to the leaf node, the calculation formula is as follows:

\[ c(\psi) = \begin{cases} 
2H(\psi - 1) - \frac{2(\psi - 1)}{n}, & \psi > 2 \\
1, & \psi = 2 \\
0, & \text{the others}
\end{cases} \]  

(2)

2.2 EasyEnsemble

The EasyEnsemble algorithm is an undersampling method used to deal with class-imbalance problems and is suitable for training ensemble models. EasyEnsemble divides the data into a majority-class dataset \( N \) and a minority-class dataset \( P \). \( N \) is undersampled multiple times independently to get \( k \) subsets of \( \{N_1, N_2, \ldots, N_k\} \). Each training uses \( N_i(1 \leq i \leq k) \) \( N_\{1,2,\ldots,k\} \) and \( P \) to train a base classifier. Overall, each base classifier was trained with a balanced training set, and the entire data was used in training without missing features.

3 Problem description and algorithm introduction

In the credit evaluation, there is the phenomenon of forging data to deceive the existing credit evaluation model. This kind of the data is not recognized by the existing model, causing serious losses to the credit evaluation work. We propose an iForest model that combines with the ideas of EasyEnsemble, called the EIF model. And design a fake data generation algorithm based on actual experience.

3.1 EIF model

As earlier, credit evaluation work is characterized by two kinds of imbalance, class-imbalance and cost-sensitive. To solve the problems caused by imbalance, we use integration learning which constructs multiple base classifiers to jointly deal with a difficult problem. Each base classifier is trained by a balanced dataset constructed under undersampling to solve the class-imbalance problem; meanwhile, each balanced training set contains all bad credit samples, and base classifier fully learns the features of the bad credit samples, and the ensemble model focuses more on the bad credit samples to optimize for the cost-sensitive problem. We call this ensemble model as enhanced isolated tree model (EIF model). Figure 1 shows the structure of the EIF model.

In ensemble learning, in order to ensure the excellent effect of the ensemble classifier, the base classifier should be "good but different". The mean is that each base classifier should have a certain classification ability and have different learning directions for the overall data. This paper adopts the sample random sampling and attribute random sampling method to generate the difference among the base classifiers. The specific process is shown in Algorithm 1:

**Algorithm 1**: EIF model

Input: train set \( D_{train} \), amount of balance set \( k \), amount of isolation tree \( \psi \), batch size \( l \), amount of attributes \( a \)

Output: EIF model \( m_{EIF} \)

1. \( N \rightarrow \) good credit part of \( D_{train} \)
2. \( P \rightarrow \) bad credit part of \( D_{train} \)
3. \( m \rightarrow \) count(\( P \))
4. for \( i=0 \) to \( k \) do
5. \( \beta = P \rightarrow \) randomly draw \( \psi \) samples out of \( N \)
6. \( \gamma = \beta \rightarrow \) randomly draw \( \psi \) attributes out of \( \beta \)
7. \( \delta = \gamma \rightarrow \) randomly draw \( l \) samples out of \( \gamma \)
8. train \( T_i \) is \( \beta \)
9. end for
10. end for
11. return model \( m_{EIF} \)
3.2 Generating anomalies

In real life, there is a phenomenon of maliciously modifying data to obtain a good credit rating. This study uses the correlation between attributes and credit to simulate deliberate fraud in credit assessment, and falsifying data set. We treat falsified data as anomalies. The process of generating anomalies is shown in the Fig. 2.

The overall idea of the algorithm is to use the correlation coefficient between each attribute and credit to forge each sample and finally get the forged samples identified as good credit by the evaluation model.

We filtered each attribute based on its correlation coefficient with credit. Algorithm calculates the Pearson correlation coefficient between each attribute \( \{A_1, A_2, \ldots, A_n\} \) and the credit \( C \), as shown in the following formula:

\[
\rho_{A_i, C} = \frac{\text{cov}(A_i, C)}{\sigma_{A_i} \sigma_C}.
\] (3)

In the experiment, we set the forged threshold value of 0.1, attributes with absolute value of correlation coefficient greater than 0.1 are considered to have falsification value. According to the Pearson correlation coefficient, a dictionary is generated as a forgery rule. In this paper, \( C = 0 \) means good credit, \( C = 1 \) means bad credit, forgery-dictionary \( F = \{f_1, f_2, \ldots, f_n\} \) generation rules such as formula (4). Attributes with low credit evaluation correlation coefficients are excluded, and attributes that are positively correlated with \( C \) are adjusted to the minimum value, otherwise, adjusted to the maximum value.

\[
f_i = \begin{cases} 
\min(A_i), & \rho_{A_i, C} > 0.1 \\
\max(A_i), & \rho_{A_i, C} < -0.1 \\
\text{remove}, & |\rho_{A_i, C}| \leq 0.1
\end{cases}
\] (4)

In the forging process of the algorithm, randomly select some bad credit samples. Each samples selects k number of attributes, and replaces these attributes by the forgery-dictionary \( F \). These samples are predicted by pretrained simple model \( F \). The samples which predict a good credit, are successful forgeries. The test set with anomalies uses forgeries to modify the original test set. The credit data forgery algorithm is as follows:

Algorithm 2. The credit data forgery
Input: dataset \( D \), amount of fake attributes \( k \), fake rate \( c \).
Output: train set \( D_{\text{train}} \), test set with anomalies \( D_{\text{test}} \).
1. \( C \leftarrow \text{Credit samples in } D \);
2. \( A \leftarrow \text{Attributes of samples in } D \);
3. \( \rho_{A_i, C} \leftarrow \text{Calculate the correlation coefficient between } A_i \text{ and } C \);
4. \( F \leftarrow \text{Initialize empty dictionary} \);
5. \( \text{foreach } \rho_{A_i, C} \text{ do} \)
6. \( \quad \text{if } \rho_{A_i, C} > 0.1 \)
7. \( \quad \quad F \leftarrow \text{Append } (A_i, \min(A_i)) \);
8. \( \quad \text{elseif } \rho_{A_i, C} < 0.1 \)
9. \( \quad \quad F \leftarrow \text{Append } (A_i, \max(A_i)) \);
10. \( \quad \text{end if} \)
11. \( \text{end for} \)
12. \( D_{\text{train}}, D_{\text{test}} \leftarrow \text{Stratification sampling in } D \);
13. \( \text{model}\_\text{init} \leftarrow \text{Initialize simple check model} \);
14. \( \text{train model in } D_{\text{train}} \);
15. \( D_{\text{train}} \leftarrow \text{Initialize empty dataset} \);
16. \( \text{while count}(D_{\text{train}}) < (c \times \text{size of bad credit sample in } D_{\text{train}}) \)
17. \( \quad x \leftarrow \text{Randomly select } k \text{ attributes and modify values by } f \);
18. \( \quad \text{if model predict } x = 0 \)
19. \( \quad D_{\text{train}} \leftarrow \text{Append } x \);
20. \( \text{D}_{\text{test}} \leftarrow \text{Drop } x \);
21. \( \text{end while} \)
22. \( \text{return } D_{\text{train}}, D_{\text{test}} \).

3.3 EIF-based anomaly credit data detection

When the credit evaluation model is applied on a data set with mixed anomalies, the model performance will drop significantly. Credit evaluation model with EIF model will judge whether each sample is anomaly through the EIF model firstly. Then, normal samples will be used for credit evaluation. This paper considers anomalies as bad credit. Anomalies can be marked, and further researched to gain more value. It is not within the scope of this study, so we leave this question aside. The workflow of EIF model is shown in Fig. 3.

4 Experiment

This section describes the environment and the data set of the experiment, and introduces the evaluation metrics that meet the requirements of the imbalance problems. The
experiment is divided into three parts to evaluate the capability of the falsification algorithm, the capability of the EIF model under different conditions, and the comparison of EIF with other anomaly detection algorithms.

### 4.1 Experimental environment

EIF model and credit data forgery algorithm are developed based on Spyder under Windows 10 system, using python programming, the hardware environment is: (AMD Ryzen 7 5800H @3.20 GHz with 16 Gbytes of RAM). The experiment was carried out in the same environment.

### 4.2 Data source and preprocessing

The dataset used in this paper is the German dataset in the UCI public database. The dataset describes 1000 loan records, 700 are “good credit” samples, and 300 are “bad credit” samples. The original data is represented by 19 different attributes. The original text numericalize data by one-hot representation to get the file called german-numeric. The german-numeric is represented by 24 different attributes. Based on german-numeric file, we set the good credit to 0, and the bad credit to 1 for the subsequent formula expression and experimental calculation.

There are dimensional differences between different attributes, which will cause various attributes to have different effects in the distance calculation. This paper contains a number of distance-based algorithms that require normalization to remove this effect. We use Min–Max normalization for preprocessing.

### 4.3 Model evaluation

This paper uses the accuracy rate (Accuracy) to evaluate the performance of the credit forgery algorithm, and uses F₁-score and Cost-sensitive error rate to evaluate the performance of credit evaluation model. To calculate these metrics, we need to define TP, TN, FP and FN as Table 1:

According to the confusion matrix, we can define Accuracy, Precision and Recall as follows:

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN},
\]

\[
\text{Precision} = \frac{TP}{TP + FP},
\]

\[
\text{Recall} = \frac{TP}{FN + TP},
\]

F₁-score is the harmonic mean of Precision and Recall. F₁-score considers both precision and recall, can reflect the generalization ability of the models. So, it is a common indicator for model evaluation in binary classification problems. The formula of F₁-score is as follows:

\[
F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.
\]

Cost-sensitive error rate is used to assess the effect of cost-imbalance. The cost-sensitive error rate is defined in terms of the cost matrix, cost matrix is shown as Table 2.

There are no lines or tables in the provided text.
In the above formula, \( f \) is the credit evaluation model, \( D^+ \) is the bad credit sample set, \( D^- \) is the good credit sample set. The total cost is the product of the number of times each error is made and the cost. Cost-sensitive error rate is the quotient of the total cost of all errors and the number of samples.

\[
E(f, D, cost) = \frac{E(f, D^+, cost_{1b}) + E(f, D^-, cost_{01})}{m},
\]

(11)

Cost-sensitive error rate sets different cost for each error, so it can reflect the overall loss of the model. Our experiments will use the cost-sensitive error rate as a measure to determine the ability of the model to face cost-sensitive problems.

### 4.4 Experimental design and results

For testing the performance of our methods, we designed a series of comparative experiments. The models and algorithms which we contract with are shown in Table 3.

| Credit evaluation model | Anomaly detection algorithm |
|------------------------|-----------------------------|
| Logistic Regression (LR) | OneClassSVM (OCSVM) |
| Naive Bayes (NB) | |
| Random Forest (RF) | Local Outlier Factor (LOF) |
| Decision Tree (DT) | |
| Support Vector Machines (SVM) | Isolation Forest (iForest) |

Firstly, we verify the effect of the fake dataset which is generated by the credit data forgery algorithm. We use five-fold cross-validation, the dataset is divided into five parts, and taking one part as test set for each experiment, the other data is train set. The algorithm modifies all bad credit samples in the test set and repeats the experiment with different amount of attributes \( k \). algorithm use LR as simple check model. We get a completely fake set. Common credit evaluation models are trained by the train set, then predict credit for fake set. We check whether the fake set fool these models. Five parallel experiments were performed, then averaging Accuracy as result.

Table 4 shows the Accuracy of various evaluation models for the recognition of fake samples generated by the forgery algorithm. The forgery algorithm falsifies samples based on the correlation coefficient between each attribute and credit, so the samples obtained by falsification have the greatest impact on the LR and SVM using the linear correlation. The reason why the algorithm is less effective in the NB and DT models is that the NB and DT models are more concerned with the effect of some characteristics.

| Fake rate | 5% | 10% | 20% | 50% |
|-----------|----|-----|-----|-----|
|            | \( F_1 \)-score | CR | \( F_1 \)-score | CR | \( F_1 \)-score | CR | \( F_1 \)-score | CR |
| LR        | 0.5295 0.8990 | 0.5181 0.9190 | 0.4777 0.9840 | 0.3412 1.1890 |
| EIF-LR    | 0.5234 0.7680 | 0.5215 0.7730 | 0.5120 0.7980 | 0.4605 0.9230 |
| change    | -0.0061 -0.1310 | 0.0034 -0.1460 | 0.0342 -0.1860 | 0.1193 -0.2660 |
| NB        | 0.5528 0.7780 | 0.5320 0.8230 | 0.5164 0.8530 | 0.4210 1.0430 |
| EIF-NB    | 0.5262 0.7180 | 0.5176 0.7430 | 0.5135 0.7530 | 0.4628 0.8880 |
| change    | -0.0266 -0.0600 | -0.0144 -0.0800 | -0.0028 -0.1000 | 0.0417 -0.1550 |
| RF        | 0.4977 0.9630 | 0.4776 0.9970 | 0.4496 1.0390 | 0.3241 1.2120 |
| EIF-RF    | 0.5066 0.8120 | 0.5059 0.8160 | 0.4949 0.8440 | 0.4490 0.9500 |
| change    | 0.0089 -0.1510 | 0.0283 -0.1810 | 0.0453 -0.1950 | 0.1249 -0.2620 |
| DT        | 0.4424 0.9980 | 0.4399 1.0080 | 0.4548 0.9810 | 0.3528 1.1650 |
| EIF-DT    | 0.4748 0.8470 | 0.4804 0.8390 | 0.4978 0.7910 | 0.4488 0.9150 |
| change    | 0.0325 -0.1510 | 0.0405 -0.1690 | 0.0431 -0.1900 | 0.0960 -0.2500 |
| SVM       | 0.4741 0.9960 | 0.4611 1.0160 | 0.4352 1.0560 | 0.3143 1.2210 |
| EIF-SVM   | 0.5023 0.8220 | 0.5006 0.8270 | 0.4944 0.8420 | 0.4479 0.9520 |
| change    | 0.0283 -0.1740 | 0.0395 -0.1890 | 0.0592 -0.2140 | 0.1336 -0.2690 |
of the sample on the whole, and changes in a few attributes may not affect the model’s judgment, even if these attributes are highly correlated with the sample. Overall, the forgery algorithm in this paper has varying degrees of deceptive effects on commonly used credit evaluation models. Therefore, it is believed that the forgery algorithm has the ability to forge samples with bad credit into good samples.

The credit data forgery algorithm selects $k = 3$, LR as check model, then generates test sets with varying amounts of fake samples. These test sets are predicted by common credit evaluation models and models with EIF. We use $F_1$-score and Cost-sensitive error rate (CR) for evaluate performance of models. In the original Cost-matrix, $\text{cost}_{01} = 1$, $\text{cost}_{10} = 5$. Set EIF model parameters after tuning: $k = 10$, $m = 10$, $l = 256$, $c = 19$. Perform a five-fold cross-validation experiment, get mean as result. The result is shown as Table 5.

The above chart shows that models with EIF can improve the $F_1$-score and obviously reduce the cost-sensitive error rate, especially in a high fake rates environment. When the percentage of forged samples is low, EIF improves the $F_1$-score of the credit evaluation model less, and even weakly reduces the $F1$ score of the model in the NB model, but the $F_1$-score is still well improved when the percentage of spurious is high. Meanwhile, the Cost-sensitive error rate of the model significantly decreased after EIF processing. EIF improve performance of credit evaluation models, and increases effects with the rise of fake rate. We can determine EIF have ability to fight against anomalies.

We also use various anomaly detection algorithm instead of EIF, test their performance in the same credit evaluation problem. We still use the credit evaluation model and test sets of the previous experiment. In this experiment, fake rate $c = 20\%$. Using $F_1$-score and Cost-sensitive error rate (CR) for contrast. The result as Table 6.

Each optimal result is highlighted in bold. We can see that most of the models get the best $F_1$-scores after EIF processing, and the cost-sensitive error rate gap is tiny compared to the best anomaly detection algorithm. The reason for the significant decreasing effect of OCSVM on cost-sensitive error rate is that the strict division bound of OCSVM classifies as many suspected fake samples as possible, which can be reflected by the significant decrease in $F_1$-score of the models using the OCSVM algorithm. LOF has a better $F_1$-scores in NB model because LOF is more “tolerant” of forged samples as long as they are partially supported by similar points, which can be seen from the high cost-sensitive error rate of LOF in all credit evaluation models. Compared with the simple IF algorithm, EIF both have higher $F_1$-score with lower cost-sensitive error rate, which can be seen as a better improvement of IF.

When ensuring lower cost loss, EIF model pay more attention to improving accuracy. So, model with EIF have the best $F_1$-score and low cost-sensitive error rate. It is proved that the EIF model has a good ability to detect anomaly in the credit data, both class-imbalance and cost-sensitive problem can be improved. EIF model is an excellent choice for anomaly in credit evaluation work.

### Table 6: Comparison of the effect of different anomaly detection algorithms of each model

|       | LR   | NB   | RF   | DT   | SVM  |
|-------|------|------|------|------|------|
|       | $F_1$-score | CR | $F_1$-score | CR | $F_1$-score | CR | $F_1$-score | CR | $F_1$-score | CR |
| vanilla | 0.4777 | 0.9840 | 0.5164 | 0.8530 | 0.4496 | 1.0390 | 0.4548 | 0.9810 | 0.4352 | 1.0560 |
| OCSVM  | 0.4839 | 0.7990 | 0.4875 | 0.7670 | 0.4765 | 0.8250 | 0.4830 | 0.7740 | 0.4786 | 0.8160 |
| LOF    | 0.4819 | 0.9750 | 0.5199 | 0.8430 | 0.4549 | 1.0300 | 0.4561 | 0.9760 | 0.4405 | 1.0470 |
| iForest| 0.4979 | 0.8170 | 0.5028 | 0.7730 | 0.4872 | 0.8480 | 0.4952 | 0.7860 | 0.4902 | 0.8390 |
| EIF    | **0.5120** | 0.7980 | **0.5135** | 0.7530 | **0.4949** | 0.8440 | **0.4978** | 0.7910 | **0.4944** | 0.8420 |

### 5 Conclusion

According to the problem that new data will be forged against the existing model in the evaluation work of credit data, this paper proposes a method to detect the fake data by using the anomaly detection model EIF based on Isolation Forest and EasyEnsemble. The EIF model trains the base model by all the minority and undersampling majority classes. The class-imbalance problem is solved without missing global data. We use linear correlations to generate fake data for evaluating models, these fake data simulate anomalies in credit data. The common credit evaluation models with EIF model are assessed by $F_1$-score and cost-sensitive error rate. The results are significantly better than vanilla model. And EIF model compared with other detection model, has better optimization effect. EIF model is effective in enhancing the performance of the credit evaluation model.

In this paper, anomalies detected are directly classified as bad credit. In the future, our directions are analyzing each credit anomaly for more information and refining credit evaluation models through abnormal information. Finally, discovering causes of anomalies from the perspective of credit evaluation work, restoring anomalies to normal credit data.
Author contribution Zhang Xiaodong: Final manuscript writing and checking; Yao Yuan: Original draft preparation; Lv CongDong: Manuscript format; Wang Tao: measurements and data base, visualization.

Funding This research was funded by National Key R&D Program of China (Grant No. 2019YFB1404602).

Declarations

Ethics approval and consent to participate Not applicable.

Consent for publication Not applicable.

Conflict of interest The authors declare no competing interests.

References

1. Ariza-Garzón MJ, Arroyo J, Caparrini A, Segovia-Vargas MJ (2020) Explainability of a machine learning granting scoring model in peer-to-peer lending. IEEE Access 8:64873–64890
2. Vojtek M, Koeenda E (2006) Credit-scoring methods. Czech Journal of Economics and Finance (Finance a uver) 56(3–4):152–167
3. Uddin MS, Chi G, Al Janabi M et al (2020) Leveraging random forest in micro-enterprises credit risk modelling for accuracy and interpretability. Int J Financ Econ 1(2):1–17
4. Chen QW, Wang W et al (2018) Class-imbalance credit scoring using Ext-GBDT ensemble. Application Research of Computers 35(2):421–427
5. Jabeur SB, Sadaaoui A, Sghaier A et al (2020) Machine learning models and cost-sensitive decision trees for bond rating prediction. Journal of the Operational Research Society 71(8):1161–1179
6. Itoo F, Singh S (2021) Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection. Int J Inf Technol 13(4):1503–1511
7. Ye XF, Lu YH (2018) Credit assessment model based on Random Forest and Naïve Bayes. J Mathematics in Practice and theory 47:68–73
8. Yu L, Yao X, Wang SY et al (2011) Credit risk evaluation using a weighted least squares SVM classifier with design of experiment for parameter selection. Expert Syst Appl 38(12):15392–15399
9. Liu Y, Yang K (2021) Credit fraud detection for extremely imbalanced data based on ensemble deep learning. Journal of Computer Research and Development 58(3):539
10. Horak J, Vrba J, Suler P (2020) Support vector machine methods and artificial neural networks used for the development of bankruptcy prediction models and their comparison. Journal of Risk and Financial Management 13(3):60
11. Le HH, Viviani JL (2018) Predicting bank failure: an improvement by implementing a machine-learning approach to classical financial ratios. Res Int Bus Financ 44:16–25
12. Ren JD, Liu XQ et al (2019) An multi-level intrusion detection method based on KNN outlier detection and random forests. Journal of Computer Research and Development 56(3):566
13. Breunig MM, Kriegel HP, Ng RT et al (2000) LOF: Identifying density-based local outliers. ACM SIGMOD Rec 29(2):93–104
14. Yang J, Rahardja S, Fränti P (2021) Mean-shift outlier detection and filtering. Pattern Recogn 115:107874
15. Campos GO, Zimek A, Sander J et al (2016) On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min Knowl Disc 30(4):891–927
16. Erfani SM, Rajasegarar S, Karunasekera S et al (2016) High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. Pattern Recogn 58:121–134
17. Liu F, Ting KM, Zhou ZH (2012) Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD) 6(1):1–39
18. Bandaragoda TR, Ting KM, Albrecht D et al (2018) Isolation-based anomaly detection using nearest-neighbor ensembles. Comput Intell 34(4):968–998
19. Fernández A, García S, Herrera F et al (2018) SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. Journal of artificial intelligence research 61:863–905
20. Liu X, Wu J, Zhou Z (2008) Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 39(2):539–550
21. Frumosu FD, Khan AR, Schiøler H et al (2020) Cost-sensitive learning classification strategy for predicting product failures. Expert Syst Appl 161:113653

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.