Journal of Computer and Communications, 2021, 9, 48-60
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Improved Isolation Forest Algorithm for Anomaly Test Data Detection

Yupeng Xu[1], Hao Dong[1,2,3*], Mingzhu Zhou[1], Jun Xing[1], Xiaohui Li[1], Jian Yu[1]

[1]China National Tobacco Quality Supervision and Test Center, Zhengzhou, China
[2]Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei, China
[3]University of Science and Technology of China, Hefei, China
Email: xuyp94@foxmail.com, *dongh@ztri.com.cn

How to cite this paper: Xu, Y.P., Dong, H., Zhou, M.Z., Xing, J., Li, X.H. and Yu, J. (2021) Improved Isolation Forest Algorithm for Anomaly Test Data Detection. *Journal of Computer and Communications*, **9**, 48-60.
https://doi.org/10.4236/jcc.2021.98004

Received: July 19, 2021
Accepted: August 15, 2021
Published: August 18, 2021

Abstract

The cigarette detection data contains a large amount of true sample data and a small amount of false sample data. The false sample data is regarded as abnormal data, and anomaly detection is performed to realize the identification of real and fake cigarettes. Binary particle swarm optimization algorithm is used to improve the isolation forest construction process, and isolation trees with high precision and large differences are selected, which improves the accuracy and efficiency of the algorithm. The distance between the obtained anomaly score and the clustering center of the k-means algorithm is used as the threshold for anomaly judgment. The experimental results show that the accuracy of the BPSO-iForest algorithm is improved compared with the standard iForest algorithm. The experimental results of multiple brand samples also show that the method in this paper can accurately use the detection data for authenticity identification.

Keywords

Isolation Forest, BPSO, K-Means Cluster, Anomaly Detection

1. Introduction

Anomaly is a data form that is different from the data characteristics under normal conditions. The study of classifying normal data and abnormal data is called anomaly detection. Anomaly detection is one of the important tasks of data mining [1]. Many scholars have put forward a series of ideas and methods in this field, forming a relatively complete system. The main anomaly detection methods can be divided into statistics-based [2], distance-based [3], density-based [4], model-based [5] [6], etc. according to the detection principle.

However, in reality, it is sometimes difficult to obtain enough anomaly data for training, and the collected data is often unbalanced, resulting in a decrease in the performance of many anomaly detection algorithms [7]. Isolation forest algorithm proposed by Liu *et al.* [8] calculates anomaly scores based on the path length obtained by the sample in multiple binary trees, and then judges the anomaly. Compared with other anomaly detection algorithms, this algorithm has linear time complexity, and the accuracy is good when there are few or missing abnormal data in the training set. Isolated forest algorithm has many practical applications, such as monitoring production abnormalities [9] [10], abnormal target detection [11], and data error detection [12] [13]. In actual work, we have collected physical and chemical cigarette testing data with many characteristic dimensions. Among them, the true samples account for the majority, and the fake samples account for a small proportion, which can be regarded as abnormal data, which meets the data requirements of the isolation forest algorithm. This paper is based on the isolation forest algorithm to verify the authenticity of the detection data. The detection data of real cigarettes and fake cigarettes have different degrees of discrimination in different feature dimensions, which leads to different detection capabilities of each tree in the isolation forest, but their weights for anomaly scores are the same, so some detection capabilities are insufficient trees which may have a negative effect on the final result and waste computing resources. This article draws on the improved isolation forest method proposed by Wu *et al.* [14], selects isolation trees with high accuracy and large differences to optimize the forest, and removes redundant isolation trees. For the anomaly scores, a threshold is needed for anomaly judgment. Based on the idea of K-means algorithm, this paper calculates the cluster centers of the abnormal scores of normal samples and abnormal samples, and judges abnormalities by distance.

2. Improved Isolation Forest

2.1. Isolation Forest

Isolation forest (iForest) algorithm is a commonly used unsupervised anomaly detection algorithm and is suitable for high-dimensional data sets. Different from other anomaly detection algorithms, iForest no longer describes normal sample points, but isolates abnormal points [15]. In iForest, anomaly points are defined as "more likely to be separated points", which can be understood as points that are sparsely distributed and far away from densely populated groups. In the feature space, sparsely distributed regions indicate that the probability of events occurring in this region is very low, so the data falling in these regions can be considered abnormal. In actual inspection work, the amount of data on fake cigarettes is very small and can be regarded as abnormal data.

iForest is integrated by multiple iTree, iTree and binary search tree have the same structure. In iTrees, the data set is randomly divided recursively until all sample points are isolated or iTrees reach the set height. Under this random

segmentation strategy, abnormal points usually have shorter paths, as shown in Figure 1, the abnormal points are quickly separated. Intuitively speaking, abnormal points are split at the lower level and the path traversed on iTrees is shorter, while the normal point is split at the higher level and the path traversed on iTrees is longer.

iForest can be divided into two steps [16]:

Training iTrees: Take samples from the train set, build isolation trees (iTrees), iTrees composition iForest.

Calculate anomaly score: Record the length of the test set samples on iTrees. Calculate the anomaly score of each sample point according to the anomaly score calculation equation.

The following is the detailed algorithm flow:

1) Training iTrees:

- Randomly select n points from the training data as sub-samples, and put them into the root node of an isolated tree;
- Randomly select a dimension, and randomly generate a cutting point p within the range of the current node data-the cutting point is generated between the maximum and minimum values of the specified dimension in the current node data;
- The p point divides the current node data space into 2 subspaces: put the points less than p on the left branch of the current node, and put the points greater than or equal to p on the right branch of the node;
- Recursively perform two steps on the left and right branch of the node, and continue to construct new leaf nodes until there is only one data on the leaf node (can no longer be cut), or the tree has grown to the set height.

2) Calculate anomaly score:

After obtaining iTrees, we can use iTrees to calculate the anomaly score. For each sample x, the result of each tree needs to be calculated comprehensively, and the anomaly score is calculated by the following equations:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{1}$$

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \tag{2}$$

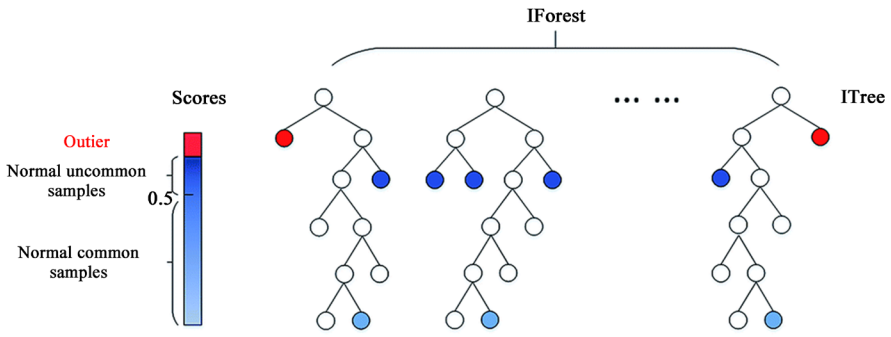

Figure 1. Normal and abnormal points in iTrees.

In Equation (1), *E(h(x))* is the average value of *h(x)* on the set of isolation trees.

$$H(k) = \ln(k) + \gamma \tag{3}$$

In Equation (3), γ is Euler constant.

h(x) is the height of *x* in each tree, and *c(n)* is the average value of the path length, which is used to standardize the path length *h(x)* of the sample *x*.

According to Equation (1):

$E(h(x)) \to c(n)$, $s \to 0.5$, the samples may not have obvious abnormal points;

$E(h(x)) \to 0$, $s \to 1$, which can be regarded as abnormal point;

$E(h(x)) \to n-1$, $n \geq 1$, $s \to 0$, can be regarded as a normal point.

According to the relationship between the number of iTrees and the average height of each sample point in the reference [8], it can be seen that when the number is selected within 10, the result is very unstable. When the number reaches 100, it tends to converge. Therefore, in this paper, the number of trees is set to 100, which can save system overhead while ensuring stable results.

2.2. BPSO

Particle Swarm Optimization Algorithm (PSO) originated from the analysis of bird foraging activities and simulates the foraging process of birds. The search space of the problem to be solved is regarded as the space where birds fly, and each bird is abstracted into a space without mass and size. Use this particle to represent a feasible solution to the problem to be solved. Therefore, the process of finding the optimal solution is equivalent to the process of foraging for birds.

In an initialized particle swarm, the position and velocity of each particle are randomly generated. The particle position represents the current solution, and the velocity represents the vector direction of the particle's current solution and the next solution. In the solution space, the particles update the speed and position of each particle according to certain rules, so that each particle moves to the best position in its own history and the best position in the global history, so as to realize the evolution of the entire population toward the optimal direction [17].

Assuming that the dimension of the search space is *D* and the number of particles in the space is *N*, the position of the particles can be expressed as a vector:

$$X_i = (x_{i1}, x_{i2}, \cdots, x_{iD}) \tag{4}$$

The flying speed of a particle can also be expressed as a vector:

$$V_i = (v_{i1}, v_{i2}, \cdots, v_{iD}) \tag{5}$$

The optimal individual extremum of the i-th particle itself can be expressed as:

$$p_{best} = (p_{i1}, p_{i2}, \cdots, p_{iD}) \tag{6}$$

The global optimal extremum of the entire particle swarm can be expressed as:

$$g_{best} = \left(g_1, g_2, \cdots, g_D\right) \tag{7}$$

In the iterative process of the algorithm, all particles update their speed and position according to the following equations:

$$v_i\left(t+1\right) = \omega v_i\left(t\right) + c_1 r_1\left(t\right)\left[p_{i,best}\left(t\right) - x_i\left(t\right)\right] + c_2 r_2\left(t\right)\left[g_{best}\left(t\right) - x_i\left(t\right)\right] \tag{8}$$

$$X_i\left(t+1\right) = X_i\left(t\right) + v_i\left(t+1\right) \tag{9}$$

In Equation (8), c_1 and c_2 are constants set by experience, r_1 and r_2 are uniformly distributed random numbers in the range [0, 1]. ω is inertia weight, the larger ω, the stronger the global search ability and the weaker the local search ability, on the contrary, the weaker the global search ability and the stronger the local search ability [18]. For the value of ω, a dynamic method can be used. At the beginning of the iteration, a larger inertia weight can be set, so that the algorithm has a strong global search capability. As the number of iterations of the algorithm increases, the inertia weight coefficient can be reduced to ensure that the particles can have a strong local search ability and perform a detailed global search around the extreme points. The value of ω in this paper is as follows:

$$\omega = \omega_{max} - \frac{\left(\omega_{max} - \omega_{min}\right)\cdot t}{T_{max}} \tag{10}$$

The set selection of iTrees in this article is a discrete problem, and the standard PSO algorithm is mainly aimed at continuous problems. For this type of discrete problem, Kennedy, J. *et al.* [19] proposed Binary Particle Swarm Optimization algorithm (BPSO).

In BPSO, the discrete space is mapped to the continuous space, and appropriate modifications are made. Still retaining the PSO's velocity-position update strategy, the value of the particle in the state space is limited to 0 or 1, and the particle velocity update equation remains unchanged.

The update method of the particle position is as follows:

$$s\left(v_{i,j}\right) = \frac{1}{1+\mathrm{e}^{\left(-v_{i,j}\right)}} \tag{11}$$

Use the sigmoid function to map the velocity of the particles to the range [0, 1], so the speed of the particle can be expressed as:

$$x_{i,j} = \begin{cases} 1 & rand\left(\right) \le s\left(v_{i,j}\right) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

2.3. BPSO-iForest

The idea of the BPSO-iForest improved algorithm is: in the iTrees set T generated in the train set D, the BPSO algorithm is used to find the optimal subset T' to meet the needs of reducing the number of base classifiers while improving the classification accuracy and execution efficiency.

Two principles are generally followed in the selection of iTree: 1) Choose

the one with higher precision; 2) Choose the one with big difference. The high-precision base classifier is selected because iForest is an integrated classifier that uses base classifier voting. The low-precision iTree may mislead the final judgment. The choice of base classifiers with large differences may complement the different information obtained between the base classifiers to increase the generalization ability of iForest.

This paper uses the cross-validation method to calculate the accuracy value of the current tree each time and takes the average accuracy value A, where the larger A represents the better the accuracy. Use Equation (8) and Equation (9) to calculate the difference between each iTree and the average difference between all iTrees [14]:

$$Q_{i,j} = \frac{n_i + n_j - 2n_{ij}}{n_i + n_j} \tag{13}$$

$$\bar{Q}_i = \frac{1}{N-1}\sum_{j=1}^{N} Q_{(i,j)} \tag{14}$$

In Equation (8), n_i and n_j represent the number of samples that are correctly tested in the sample space D of the tree T_i and T_j, respectively, and n_{ij} represents the number of samples that are correctly tested by T_i and T_j.

According to reference [14], find a fitness function that can balance these two factors as shown in Equation (15):

$$F(T') = \frac{1}{\mu\bar{A} + \lambda\bar{Q}'} \tag{15}$$

μ and λ and represent the weight of A and Q.

The specific algorithm flow is as follows:

Input: Train set D; Number of particles N; Number of iterations C;

Step 1. Build iTrees-Initial quantity L;

Step 2. Initialization: particle velocity, position, p_{best} and g_{best};

Step 3. Calculate particle $\bar{A}$, $\bar{Q}'$ and $F(T')$;

Step 4. Update particle swarm's p_{best} and g_{best};

Step 5. Update ω;

Step 6. Update particle velocity and position;

Step 7. If the extreme value of the particle swarm reaches the optimal solution or the number of iterations > C, the algorithm ends. Otherwise repeat step 2;

Output: The optimal subset T'.

3. Threshold Selection

According to the principle of iForest, when the abnormal score $s \to 1$, the sample is an abnormal point; $s \to 0$, the sample is a normal point. The value of the abnormal score is within (0,1), and a threshold must be determined to divide the abnormal score into an abnormal interval or a normal interval. The distribution of abnormal scores for samples of different brands is different, so the specific value of the threshold cannot be fixed in the algorithm. Therefore, it is necessary

to design a method to calculate the threshold based on the result of the anomaly score. Combined with the idea of K-Means clustering, this article came up with a clustering method to calculate the center points center 0 and center 1 of the two types of abnormal scores. According to the distance between the abnormal score and the center points, it is divided into normal or abnormal.

K-Means is a commonly used clustering algorithm based on Euclidean distance, the main purpose is to divide n sample points into k clusters, so that similar samples can be divided into the same cluster as much as possible. The K in the K-means algorithm represents data that needs to be divided into several categories, which are manually defined. All samples in this article are divided into abnormal and normal categories, only two center points need to be found, so the number of cluster centers is k = 2. The location of K-Means initialization centroid has a great impact on the final clustering results and running time. If the initial centroid is not well chosen, it will fall into a local optimal solution, so it is necessary to choose a proper initial centroid position. Combined with the idea that the distance between the initial centroids of the K-Means++ improved algorithm should be as large as possible [20] and the characteristics of low abnormal scores of normal points and high abnormal scores of abnormal points, the maximum and minimum anomaly scores are directly set as the initial class center points, which can speed up the convergence of the algorithm.

Algorithm steps:

1) Select the minimum and maximum anomaly scores as the initial cluster centers center0 and center1;

2) For each sample's anomaly score $score_i$, calculate its distance to the two cluster centers, and assigns them to the clusters corresponding to the closer cluster centers;

3) Recalculate the values of center0 and center1, and update the centroids of all samples belonging to the class. The update equation of the cluster centers is as shown in Equation (16);

$$center = \frac{1}{|c_i|}\sum_{x \in c_i} x \tag{16}$$

4) The algorithm terminates when the function reaches the number of iterations or the minimum error change. If the conditions are not met, steps 2 and 3 are repeated.

The specific algorithm flow is shown in **Algorithm 1**.

In **Algorithm 1**, $\Delta J = |center - center'|$, δ is minimum error change, center 0 is the center of the anomaly, center 1 is center of normal class. When classifying the test set, first calculate samples' anomaly scores. With the class centers center 0 and center 1, for a new unknown sample, divide it into closer classes.

4. Experimental Results and Analysis

4.1. Lab Environment

In order to verify the accuracy of the algorithm's division of abnormal data, this

Algorithm 1. *ClassifyByCluster(scores, iters)*

Inputs: scores - anomaly scores, iters - number of iterations
Output: anomaly class centers

```
1. center0 = min(scores)
2. center1 = max(scores)
3. labels [scores.length]
4. for n = 1 to iters
5.         initialization cnt0,cnt1
6. for i =1 to scores.length
7.                 diff0 = Math.abs(scores [i]-center0)
8.                 diff1 = Math.abs(scores [i]-center1)
9.                 diff0 < diff1? labels [i] = 0&cnt0++:labels [i]=1&cnt1++
10. end
11. diffs = centers
12. initialization centers
13. update centers
14. if Δf<δ
15. break
16.         end
17. return center0,center1
```

paper uses cigarette physics and chemistry inspection data to test the algorithm. Cigarette physical and chemical inspection data include: length mean and standard deviation, circle mean and standard deviation, mass mean and standard deviation, resistance to suction mean and standard deviation, hardness mean and standard deviation, ventilation rate mean and standard deviation, tar amount, carbon monoxide amount and nicotine amount, a total of 15 feature dimensions of detection data. Collected sample data of multiple brand specifications, among which the real sample data includes data of different years and different production batches. In order to ensure the general applicability of the algorithm, the inspection data comes from different laboratories and randomly select the train set and the test set.

4.2. Result Analysis

Take brand 1 as an example, real cigarette samples are regarded as normal points, fake cigarette samples are regarded as abnormal points. The train set has a total of 229 samples, including 216 normal samples, 13 abnormal samples, the test set has a total of 58 samples, including 46 normal samples and 12 abnormal samples. Build iTrees with train set, get the test set data's abnormal score on the iTrees, draw a scatter plot of abnormal scores, with orange for abnormal samples and blue for normal samples. Figure 2 is the experimental result of BPSO-iForest, Figure 3 is the experimental result of iForest.

Calculate the two cluster centers by **Algorithm 1**, evaluate the correct rate of judgment according to the discrimination criteria. Count the distribution range

of abnormal scores of abnormal samples and normal samples. All analysis conclusions are shown in Table 1.

It can be seen from the experimental results that, compared with iForest, BPSO-iForest has higher accuracy; the distribution range of abnormal scores becomes smaller, abnormal samples:0.29 → 0.09, normal samples:0.14 → 0.10.

In actual work, the abnormal sample data of the train set is often missing or insufficient. In order to evaluate the accuracy of the algorithm when the train set lacks abnormal data, the selected train set lacks abnormal samples, only 174

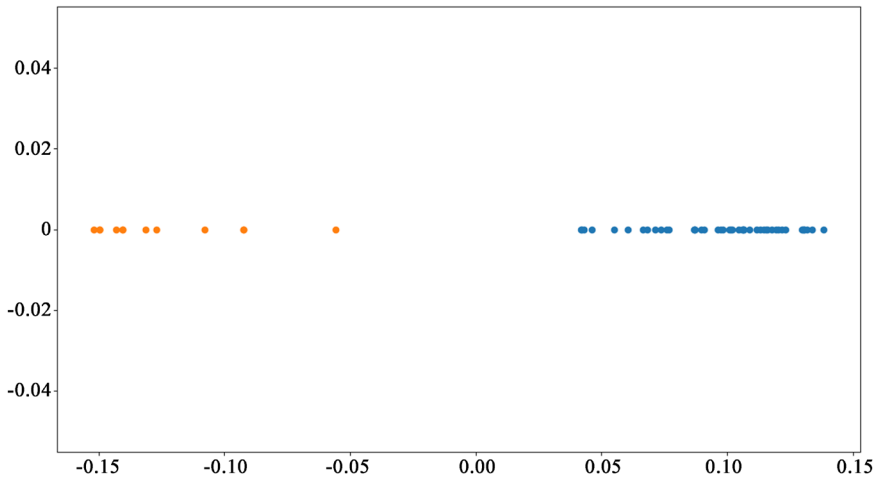

Figure 2. BPSO-iForest scatter plot.

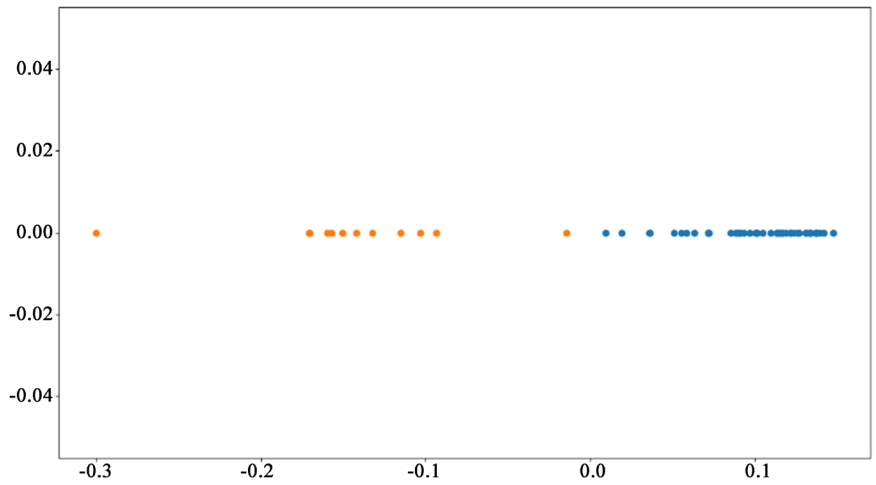

Figure 3. iForest scatter plot.

Table 1. BPSO-iForest and iForest experimental results.

Project	BPSO-iForest	iForest
center 0 *center* 1	−0.12, 0.10	−0.15, 0.10
Abnormal samples score range	(−0.15, −0.06)	(−0.30, −0.01)
Normal samples score range	(0.04, 0.14)	(0.01, 0.15)
Correct rate	100%	98.3%

normal samples, test set include 40 normal samples and 6 abnormal samples. Figure 4 is the experimental result.

The two cluster centers calculated by **Algorithm 2** are −0.06 and 0.15. According to the discrimination criteria, the accuracy of the judgment is 100%. The range of abnormal scores for real samples is (0.06, 0.23), larger than when there are abnormal samples in the train set. In the absence of train set's abnormal samples, the distribution of abnormal scores is scattered, which is consistent with the calculation principle of isolation forest algorithm. In the case of missing abnormal data in the train set, the degree of difference of iTrees is reduced, resulting in a larger range of scores.

Comparison between normal samples at different production times. Try to compare the control of physical and chemical data between different batches by the abnormal scores. The train set include 87 normal samples produced in 2020, 4 abnormal samples were added as a reference, and 20 normal samples produced in 2018. Figure 5 is the experimental result.

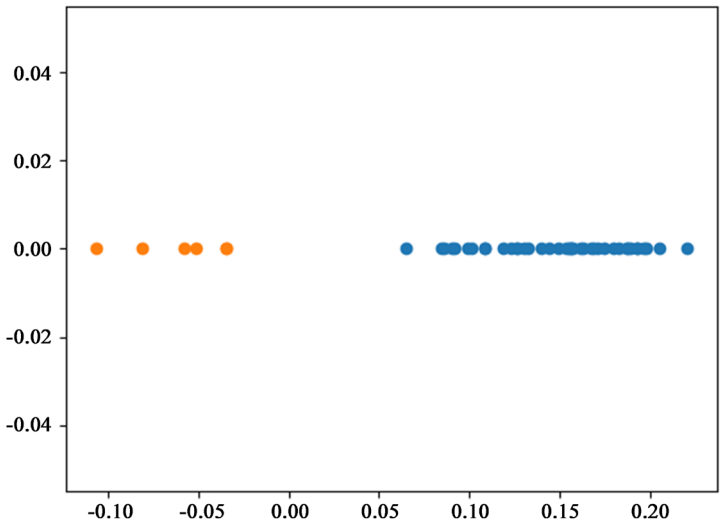

Figure 4. Abnormal data missing from the training set scatter plot.

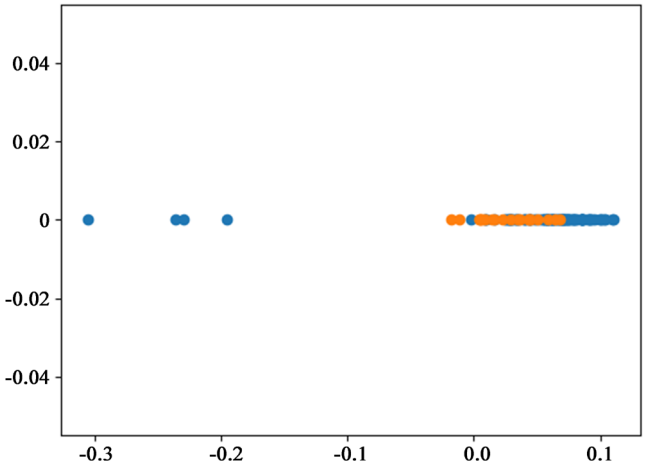

Figure 5. Comparison of different batches scatter plot.

On the left are the 4 abnormal samples, the orange on the right are 2018 samples, and the blue are 2020 samples. The 2018 samples and 2020 samples have a high degree of overlap in the abnormal score distribution, which are all clearly distinguished from the abnormal samples. The above results show that there is no significant difference between the physical and chemical data produced in 2018 and 2020, and the product quality stability is well controlled.

In order to verify the accuracy of the algorithm on other brand products, this paper selected ten other brands for experimentation. All results are shown in Table 2.

It can be seen from Table 2 that the method in this paper has generally better identification accuracy for different brand products.

5. Conclusions

In order to identify and detect fake or defective cigarettes through physical and chemical test data, this article uses BPSO-iForest algorithm to calculate the anomaly score of samples and K-Means algorithm to get the cluster center, finally determine whether the sample data is abnormal.

This paper uses the BPSO-iForest algorithm to find the most significant subset T in iTrees, which meets the requirements of reducing the number of base classifiers and improving classification accuracy and execution efficiency. With the abnormal score obtained by the BPSO-iForest algorithm, we can use the K-Means algorithm to obtain the cluster centers of the normal data and the abnormal data as discrimination threshold, and judge the abnormal state by the distance between the sample score and the cluster centers. Experiments were carried out on multiple brands' cigarette products test data. The experimental results show that the method in this paper can effectively identify genuine and fake cigarette samples. At the same time, taking the Brand 1 specification as an example, the BPSO-iForest and iForest algorithms are compared. The improved BPSO-iForest algorithm can improve the accuracy of the algorithm, reduce the

Table 2. 10 brands experimental results. R for real, F for fake.

Brand number	Train set R/F	Test set R/F	Correct rate R/F
1	737/54	150/12	100%/100%
2	225/25	108/31	100%/100%
3	143/15	72/10	100%/100%
4	380/31	73/8	100%/100%
5	86/10	44/25	100%/96%
6	144/13	65/7	100%/100%
7	108/10	43/31	100%/100%
8	82/10	44/14	100%/100%
9	272/29	68/8	100%/100%
10	547/30	104/10	100%/100%

interference of redundant features on abnormal scores, and improve the time efficiency of the algorithm. The lack of abnormal data in the train set is also analyzed, and the test results show that the improved BPSO-iForest algorithm can still effectively identify abnormal data.

Although the algorithm has achieved the expected results, there are still deficiencies in data mining. The contribution rate of each feature dimension to the detection result is a promising research direction, and we will continue to explore the meaning behind the test data.

Conflicts of Interest

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

References

[1] Aggarwal, C.C. (2013) Outlier Analysis. Springer, New York. https://doi.org/10.1007/978-1-4614-6396-2

[2] Wang, C., Viswanathan, K., Choudur, L., Talwar, V., Satterfield, W. and Schwan, K. (2011). Statistical Techniques for Online Anomaly Detection in Data Centers. 12*th IFIP/IEEE International Symposium on Integrated Network Management* (*IM* 2011) *and Workshops*, Dublin, 23-27 May 2011, 385-392. https://doi.org/10.1109/INM.2011.5990537

[3] Knorr, E.M. and Ng, R.T. (1998) Algorithms for Minning Distance Based Outliers in Large Data-Sets. *Proceedings of the* 24*th International Conference on Very Large Databases*, New York, 24-27 August 1998, 392-403.

[4] Breuning, M.M., Kriegel, H.P., Ng, R.T. and Sander, J. (2000) LoF: Indentifying Density-Based Local Outliers. *ACM SIGMOD Record*, **29**, 93-104. https://doi.org/10.1145/335191.335388

[5] He, Z., Xu, X. and Deng, S. (2003) Discovering Cluster-Based Local Outliers. *Pattern Recognition Letters*, **24**, 1641-1650. https://doi.org/10.1016/S0167-8655(03)00003-5

[6] Rousseeuw, P.J. and Driessen, K.V. (1999) A Fast Algorithm for the Minimum Covariance Determinant Estimator. *Technometrics*, **41**, 212-223. https://doi.org/10.1080/00401706.1999.10485670

[7] He, H. and Garcia, E.A. (2009) Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, **21**, 1263-1284. https://doi.org/10.1109/TKDE.2008.239

[8] Liu, F.T., Ting, K.M. and Zhou, Z.H. (2008) Isolation Forest. 2008 8t*h IEEE International Conference on Data Mining*, Pisa, 15-19 December 2008, 413-422. https://doi.org/10.1109/ICDM.2008.17

[9] Susto, G.A., Beghi, A. and McLoone, S. (2017) Anomaly Detection through On-Line Isolation Forest: An Application to Plasma Etching. 2017 28*th Annual SEMI Advanced Semiconductor Manufacturing Conference* (*ASMC*), Saratoga Springs, 15-18 May 2017, 89-94. https://doi.org/10.1109/ASMC.2017.7969205

[10] Zou, Z., Xie, Y., Huang, K., Xu, G., Feng, D. and Long, D. (2019) A Docker Container Anomaly Monitoring System Based on Optimized Isolation Forest. *IEEE Transactions on Cloud Computing*, p. 1. https://doi.org/10.1109/TCC.2019.2935724

[11] Huang, Y., Xue, Y. and Li, P. (2021) Subspace Analysis Forest for Hyperspectral Anomaly Detection. *Acta Geodaetica et Cartographica Sinica*, **50**, 416-425.

[12] He, Y., Zhu, X., Wang, G., Sun, H. and Wang, Y. (2017) Predicting Bugs in Software Code Changes Using Isolation Forest. 2017 *IEEE International Conference on Software Quality*, *Reliability and Security* (*QRS*), Prague, 25-29 July 2017, 296-305. https://doi.org/10.1109/QRS.2017.40

[13] Calheiros, R.N., Ramamohanarao, K., Buyya, R., Leckie, C. and Versteeg, S. (2017). On the Effectiveness of Isolation-Based Anomaly Detection in Cloud Data Centers. *Concurrency and Computation*: *Practice and Experience*, **29**, Article No. e4169. https://doi.org/10.1002/cpe.4169

[14] Wu, Z. and Zhang, S. (2021) Improved Isolation Forest Method for WSN Anomaly Data Detection. *Journal of Chinese Computer Systems*, **42**, 127-131.

[15] Ding, Z. and Fei, M. (2013) An Anomaly Detection Approach Based on Isolation Forest Algorithm for Streaming Data Using Sliding Window. *IFAC Proceedings Volumes*, **46**, 12-17. https://doi.org/10.3182/20130902-3-CN-3020.00044

[16] Liu, F.T., Ting, K.M. and Zhou, Z.-H. (2012) Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data*, **6**, Article No. 3. https://doi.org/10.1145/2133360.2133363

[17] Mannila, H., Toivonen, H. and Verkamo, A.I. (1997) Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery*, **1**, 259-289. https://doi.org/10.1023/A:1009748302351

[18] Jin, N. and Rahmat-Samii, Y. (2007) Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations. *IEEE Transactions on Antennas and Propagation*, **55**, 556-567. https://doi.org/10.1109/TAP.2007.891552

[19] Kennedy, J. and Eberhart, R.C. (1997) A Discrete Binary Version of the Particle Swarm Algorithm. 1997 *IEEE International Conference on Systems*, *Man*, *and Cybernetics*. *Computational Cybernetics and Simulation*, Orlando, 12-15 October 1997, 4104-4108. https://doi.org/10.1109/ICSMC.1997.637339

[20] Arthur, D. and Vassilvitskii, S. (2006) k-Means++: The Advantages of Careful Seeding. Stanford, New Orleans.