ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20210608123522
Original scientific paper

A Novel Oversampling Method for Imbalanced Datasets Based on Density Peaks Clustering

Jie CAO*, Yong SHI

Abstract: Imbalanced data classification is a major challenge in the field of data mining and machine learning, and oversampling algorithms are a widespread technique for re-sampling imbalanced data. To address the problems that existing oversampling methods tend to introduce noise points and generate overlapping instances, in this paper, we propose a novel oversampling method based on density peaks clustering. Firstly, density peaks clustering algorithm is used to cluster minority instances while screening outlier points. Secondly, sampling weights are assigned according to the size of clustered sub-clusters, and new instances are synthesized by interpolating between cluster cores and other instances of the same sub-cluster. Finally, comparative experiments are conducted on both the artificial data and KEEL datasets. The experiments validate the feasibility and effectiveness of the algorithm and improve the classification accuracy of the imbalanced data.

Keywords: classification; density peaks clustering; imbalanced datasets; over sampling

1 INTRODUCTION

Imbalanced datasets occur when there are significantly more instances from some classes than others. More precisely, in binary class problem, the class that consists of a large number of instances is referred to as the majority class whereas the class that consists of only a few instances is called the minority class [1]. The imbalanced dataset problem is prevalent in various real-world applications which include medical diagnosis [2], fault detection [3], credit assessment [4], etc. Among them, the minority class is the focus of identification and will bring significant losses if misclassified. For example, medical diagnoses in which a patient has a malignant disease fall into a minority class. If a malignant disease is misdiagnosed as benign, the best time to treat it will be missed and, in serious cases, the patient will die. Therefore, the effective improvement of the classification performance of unbalanced datasets is currently a hot research topic.

Due to its difficulty and prevalence, the imbalanced dataset problem has attracted much attention of researchers in the last two decades and numerous approaches have been proposed to address it, which are primarily divided into two categories: algorithm level approaches and data level approaches [5]. Algorithm level approaches focus on the introduction of cost-sensitive factors, integrated learning and other means to make the classifier more biased towards a few classes. Data level approaches seek to rebalance the class distribution by some resampling techniques including over-sampling, under-sampling and the combination of the two methods. All three methods are used to achieve a balanced set of instances by increasing or decreasing the number of instances from a particular class. In this paper, we deal with imbalanced data at the data level by increasing the number of instances in minority class.

Random replication of minority instances, although fast enough to increase the number of minority instances, is not sufficiently effective in practice and tends to lead to over-fitting. The Synthetic Minority Oversampling Technique (SMOTE) algorithm is currently the most classical oversampling algorithm, where the new instances are generated by linearly interpolating a randomly selected minority instance and one of its k nearest minority neighbors instead of merely duplicating existing instances, where k is a user-specified variable [6]. However, this approach may produce some noisy instances that overlap between the minority and majority classes because of its sensitivity to k values, and the newly generated instances may be included in the majority class, which may further degrade the classification performance of subsequent classifiers. In order to eliminate its shortcomings, numerous extensions of SMOTE have been proposed, mainly based on distance or clustering aspects. The Adaptive Synthetic sampling Technique (ADASYN) algorithm takes into account the distribution of the instance, where the minority instances close to the class boundary with more majority neighbors in their neighborhood are assigned higher weights so as to have greater chances to be oversampled [7]. However, this method still results in synthetic instances falling within the majority class region. In contrast, clustering-based oversampling methods first cluster minority instances and then generate a specific number of new instances in each class cluster to avoid generating noise across the boundary. Clustering methods include k-means clustering [8], hierarchical clustering [9], etc. However, these methods are computationally complex, requiring manual setting of clustering parameters in advance, and are very difficult for handling datasets with unknown distribution.

To address these issues, this paper proposes an improved SMOTE algorithm based on density peaks clustering (DP-SMOTE). Firstly, in order to avoid determining the clustering parameters and to handle various types of data efficiently, density peaks clustering is used to cluster a small number of classes of instances. Secondly, oversampling weights are assigned according to the inverse of the number of instances within a sub-cluster, giving higher oversampling weights to sub-clusters with a small number of instances. Thirdly, the interpolation formula is improved to perform linear interpolation between cluster cores and other instances in the same sub-cluster to avoid marginalization of the generated instances. Finally, comparison experiments are conducted on the artificial dataset and the KEEL dataset. The experimental results show the feasibility and superior performance of the

method proposed in this paper, which improves the classification accuracy of the unbalanced dataset.

2 DENSITY PEAKS CLUSTERING

Density peaks clustering (DPC) algorithm [10] is a relatively new density-based clustering algorithm proposed in recent years, whose main idea is to find high-density regions separated by low-density regions. DPC is based on two intuitive and simple assumptions that the cluster centers should have the highest local densities and simultaneously a relative larger distance to other cluster centers. Unlike *k*-means, DPC does not require the number of clusters in advance and also does not impose convex assumption on the data, which make it more preferable for arbitrarily shaped unsupervised clustering issues without any priori cluster structure information available. From its two assumptions, we can find that in order to accomplish the clustering task, DPC requires to introduce two important quantities for each instance, i.e., the local density ρ_i and the distance δ_i to the nearest neighbor with higher local density.

Specifically, we assume that the data set is $X_m = \{x_1, x_2, ..., x_m\}$, where m denotes the number of instances. For each instance x_i, its local density ρ_i can be computed via Gaussian kernel function, which is defined by:

$$\rho_i = \sum_{j \neq i} \mathrm{e}^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{1}$$

where d_{ij} is the distance between two instances x_i, x_j, and d_c is referred to be cut-off distance. d_c is usually set to be the distance in which 2% instances are included on average [9].

Further, the distance δ_i between instance x_i and higher local density neighbor is defined by:

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} \left(d_{ij}\right) \\ \max\limits_{j:\rho_j < \rho_i} \left(d_{ij}\right) \end{cases} \tag{2}$$

Density peaks clustering defines instances with both higher local densities ρ_i and larger distances δ_i as cluster centers. Instances with lower local densities ρ_i and larger distances δ_i are defined as outliers. The remaining instances are then assigned to the class clusters where the neighboring clustering centers with higher local densities are located, obtaining the corresponding class cluster labels.

3 OVERSAMPLING ALGORITHM BASED ON DENSITY PEAKS CLUSTERING

When traditional oversampling algorithms deal with minority instances, they focus more on the inter-class imbalance problem, i.e. the imbalance between minority instances and majority instances. Ignoring the influence of intra-class imbalance, the generated instance distribution often suffers from noise and marginalization, which does not conform to the original instance distribution pattern. To address the above problems, this paper proposes an oversampling algorithm based on density peaks clustering (DP-SMOTE), and the algorithm flowchart is shown in Fig. 1.

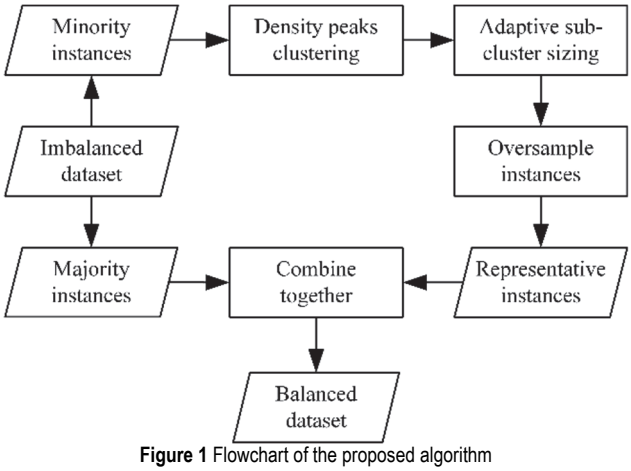


Figure 1 Flowchart of the proposed algorithm

3.1 Density Peaks Clustering for Minority Class

The process of clustering density peaks for minority instances is broken down into a number of steps. Firstly, the local density of each instance point is calculated. Secondly, the distance between each instance point and the nearest high density point is calculated. Thirdly, a decision diagram is drawn using the local density as the horizontal coordinate and the distance as the vertical coordinate, from which the clustering centers and outliers are determined. Finally, the corresponding sub-cluster labels are assigned according to the proximity of the remaining instances to the cluster centers with high local densities. The specific process is shown in Algorithm 1.

Algorithm 1 Density peaks clustering algorithm

Inpute: D_s: the set of minority instances,
d_c: the cut-off distance.
Output: $D_{s,k}$: the set of instances in each sub-cluster,
k: number of sub-clusters,
D_{center}: the set of cluster centers,
D_{noise}: the set of outlier points.
1: Determine the cut-off distance d_c, and calculate ρ_i and δ_i for each instance according to Eq. (1) and Eq. (2).
2: Draw a decision diagram based on ρ_i and δ_i, determine the cluster centers and outliers, and obtain D_{center}, D_{noise}, and k.
3: Assign the remaining instances to clustering centers with higher local density neighborhoods and divide them into k classes to obtain $D_{s,k}$.

3.2 Improved Oversampling Algorithm

After clustering a small number of classes of instances, different oversampling weights $W_{s,i}$ are first assigned according to the number of instances in each sub-cluster, which is defined by:

$$W_{s,i} = 1 \Big/ \sum_{i=1}^{k} n(i) \tag{3}$$

where k denotes the number of sub-clusters and $n(i)$ denotes the number of instances in the i-th sub-cluster. From Eq. (3), it can be seen that the oversampling weight

is inversely proportional to the number of instances in the sub-clusters, i.e. the smaller the number of instances in a sub-cluster, the larger the oversampling weight, and the more instances need to be generated to compensate for the small number of inter-class imbalances.

Then, the number of instances to be generated for each sub-cluster N_i is calculated based on the oversampling weight $W_{s,i}$, which is defined by:

$$N_i = N \times \left(W_{s,i} \Big/ \sum_{i=1}^{k} W_{s,i} \right) \tag{4}$$

where N denotes the overall number of synthetic minority instances to be generated. Generally, N is set to be the difference of majority size and minority size, which can balance the dataset with a 1:1 ratio so that both classes are of the same size.

Finally, to avoid marginalizing the distribution of the generated instances, the concept of cluster cores is introduced in this paper. Cluster cores refer to the cluster centers of each sub-cluster obtained by clustering minority classes by density peaks. Linear interpolation is performed between the cluster cores of the same sub-cluster and other instances to achieve high quality generated instances. The interpolation formula is defined as follows:

$$x_{new,i} = D_{center,i} + a \times \left(D_{center,i} - x_{ij} \right) \tag{5}$$

where a denotes a random number between (0, 1), x_{ij} denotes the j-th instance in the i-th sub-cluster, $x_{ij} \in D_{s,i}$.

In addition, outlier points are not involved in the generation of new instances to prevent noise generation.The detailed procedure of the improved oversampling procedure is shown in Algorithm 2.

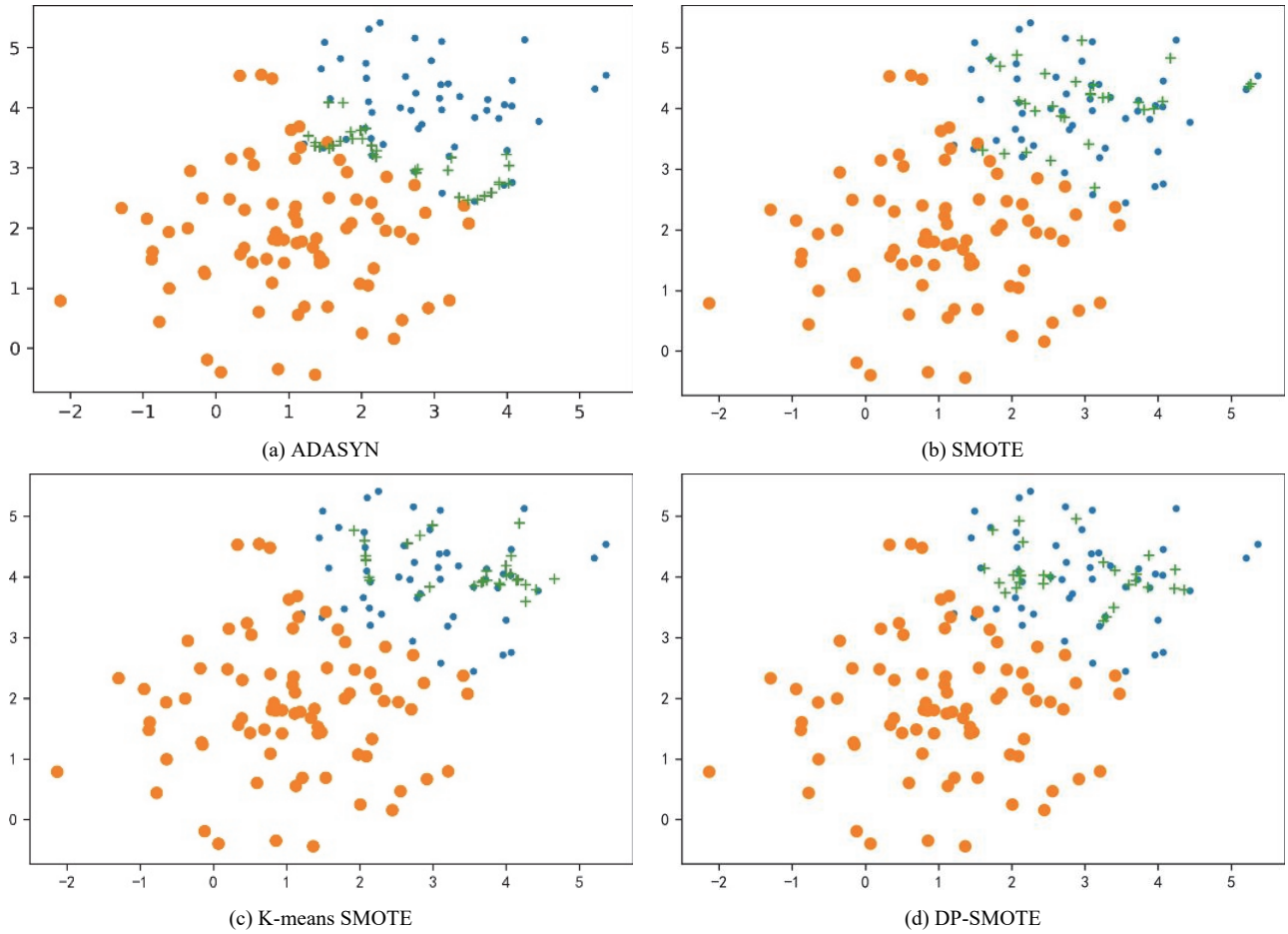
Algorithm 2 DP-SMOTE algorithm

Inpute: D: the training set.
Output: X_{new}: the set of new instances.
1: Clustering minority instances using Algorithm 1, obtain D_{center}, D_{noise}, $D_{s,k}$ and k.
2: Calculate the number of new instances N.
3: Calculate the oversampling weights $W_{s,i}$ for each sub-cluster and determine the number of oversamples for each sub-cluster N_i according to the weights.
4: Within each sub-cluster a roulette wheel is used for the selection of instances within the cluster, and new instances are generated between the cluster core and the selected instances of that sub-cluster according to Eq. (5), and this step is repeated until the number of new instances synthesized in each sub-cluster is equal to N_i.
5: Combine the sub-clusters to generate new instances and obtain X_{new}.

4 EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Artificial Dataset

In order to verify the effectiveness and superiority of the proposed method, artificial datasets are constructed to compare the distribution of instances generated by different oversampling algorithms.



(a) ADASYN (b) SMOTE

(c) K-means SMOTE (d) DP-SMOTE

Figure 2 Distribution of synthetic instances on artificial datasets with different oversampling methods

This experiment invokes the Sklearn package in the Python library to randomly generate two sets of instance sets that conform to a Gaussian distribution. The minority instances are represented by small dots and the number is 50, the majority instances are represented by large dots and the number is 80, and the newly generated minority instances are represented by plus signs and the number is 30, i.e. the difference between majority size and minority size. In terms of algorithm implementation, the SMOTE and *K*-means SMOTE algorithms are implemented by directly calling the Python imbalance-learn package, and the proposed algorithm is written in the Python language with the cut-off distance d_c taken to be located at 2%. The distribution of new instances synthesized by different oversampling algorithms is given in Fig. 2.

Fig. 2a shows the distribution of instances synthesised by the ADASYN algorithm, where some instances synthesised by the algorithm are concentrated in the boundary region between the minority and majority categories, introducing a significant amount of noise. Fig. 2b shows the distribution of the instances synthesized by the SMOTE algorithm, from which it can be visualized that some of the instances synthesized by the algorithm are scattered in the boundary region between the minority and majority classes, introducing noise. Fig. 2c shows the distribution of instances synthesized by the *K*-means SMOTE algorithm, and it can be seen that the instances synthesized by this algorithm are more densely distributed with more overlapping instances. Fig. 2d shows the distribution of new instances synthesized by the proposed algorithm, and by comparison it can be observed that the proposed algorithm generates new instances more evenly distributed in a few regions with low class density and fewer overlapping instances.

4.2 KEEL Datasets

In order to further verify the effectiveness of the method in this paper, the SMOTE, *K*-means SMOTE, ADASYN and the proposed algorithm were used to oversample minority instances for each of the seven KEEL datasets, and then the sampled data are classified using Support Vector Machine (SVM), K-Nearest Neighbor (KNN) and Gradient Boosted Decision Tree (GBDT).

Table 1 Information on the KEEL imbalance dataset

Datasets	Attribute	Instances	Minority	Majority	Imbalance ratio
Pima	8	768	268	500	1,87
Vehicle1	18	846	217	629	2,90
Ecoli2	7	336	52	284	5,46
Ecoli4	7	336	20	316	15,80
Yeast3	8	1484	163	1321	8,10
Yeast4	8	1484	51	1433	28,10
Yeast5	8	1484	40	1440	36,00

Specific information for all KEEL datasets is shown in Tab. 1, where the imbalance ratio is the ratio of the majority class to the minority class in the sample.The eigenvalues of all instances are subjected to maximum-minimum normalization. 10-fold cross validation was used and the average value is taken as the experimental result to avoid randomness influences on the results. The proposed algorithms are implemented in Python, with cut-off distance d_c taken to be located at 2%. SMOTE and *K*-means SMOTE, ADASYN are directly called from the Python imbalance-learn package. SVM, KNN, GBDT are directly called from the Python Sklearn module, all using default parameters.

4.3 Performance Evaluation

For imbalanced data, the correct classification of minority class is particularly important, and traditional evaluation methods based on a single accuracy rate are no longer applicable. Therefore, in order to quantitatively assess the classification performance of unbalanced data, in this paper, *F*-Measure, *AUC*, and *G*-Mean are used as the performance measures to compare different methods. These performance metrics usually depend on the confusion matrix, as shown in Tab. 2.

Table 2 Confusion matrix

Predict	True	
	Positive	Negative
Positive	TF	FP
Negative	FN	TN

F-Measure is a weighted summed average of Precise and Recall. The equilibrium point is achieved when both are maximized at the same time. *F*-Measure is defined by:

$$F\text{-Measure} = \frac{2\text{Precision} \times \text{Rec all}}{\text{Precision} + \text{Rec all}} \tag{6}$$

G-Mean represents the geometric mean of the accuracy of the minority and majority class instances. The maximum value is reached when and only when the predictive accuracy of the two classes of instances is balanced and can be used to evaluate the classification performance of imbalanced data. *G*-Mean is defined by:

$$G\text{-Mean} = \sqrt{\frac{TP \cdot TN}{(TP + FN) \cdot (TN + FP)}} \tag{7}$$

Area Under Curve (*AUC*) indicates the probability that a classifier will rank positive instances from a random test higher than negative instances from a random test. The value ranges from 0,5 to 1. The higher the value, the better the differentiation ability of the classifier. *AUC* is defined by:

$$AUC = \frac{1 + \text{Recall} - FP}{2} \tag{8}$$

4.4 Experimental Results and Discussion on KEEL Datasets

Tab. 3 to Tab. 5 list the *F*-Measure, *AUC* and *G*-mean obtained after processing with different oversampling algorithms using SVM, KNN and GBDT classification respectively, with the optimal data marked in bold. For a clearer visual comparison of the various methods, Fig. 3 to Fig. 5 present the average values of each evaluation metric obtained by the four oversampling algorithms on the same classifier over all datasets.

From the results in Tab. 3 to Tab. 5, we can see that DP-SMOTE achieves optimal results for all the indicators *F*-Measure in all datasets under the three classifiers, indicating that this method effectively improves the classification accuracy of a few classes. In most of the data sets, the best results were obtained for the indicators *AUC* and *G*-mean, indicating that the proposed method effectively improved the overall classification performance of the unbalanced data sets. In the three datasets Vehicle1, Ecoli2 and Ecoli4, DP-SMOTE achieves the optimal *F*-Measure and *AUC* and *G*-mean at the same time. In general, DP-SMOTE takes into account the intra-class imbalance and cross-regional noise problems, and improves the classification performance of imbalanced data.

Fig. 3 to Fig. 5 clearly demonstrate that, compared to the other three oversampling algorithms, the best classification performance is obtained by all three classifiers on the dataset obtained from the processing of the oversampling algorithm proposed in this paper, with the best average values of the evaluation metrics *F*-Measure and *AUC* and *G*-mean, which indicates that DP-SMOTE has the best average performance and has obvious advantages in dealing with the classification problem of imbalanced data.

Table 3 Result of the oversampling methods on all used datasets using SVM

Datasets	SMOTE			K-means SMOTE			ADASYN			DP-SMOTE		
	F-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M
Pima	0,6153	0,7110	0,6707	0,5522	0,7170	0,5888	0,6250	0,7120	0,6845	**0,6405**	**0,7355**	**0,6874**
Vehicle1	0,6323	0,7231	0,7756	0,5500	0,7079	0,6527	0,6466	0,7492	0,7579	**0,6788**	**0,7526**	**0,8523**
Ecoli2	0,8750	0,9058	0,9477	0,903	0,9316	0,9516	0,7568	0,8119	0,9220	**0,9333**	**0,9609**	**0,9565**
Ecoli4	0,7999	0,8581	0,9240	**0,8889**	**0,9390**	**0,9334**	0,7619	0,8277	0,9192	**0,8889**	**0,9390**	**0,9334**
Yeast3	0,7748	0,8268	**0,9394**	0,8119	0,8663	0,9230	0,7097	0,7793	0,9377	**0,8283**	**0,8804**	0,9259
Yeast4	0,2059	0,5563	**0,7428**	0,2713	0,5796	0,8107	0,1818	0,5478	0,7392	**0,2553**	**0,5772**	0,7090
Yeast5	0,5714	0,7097	0,9131	0,5490	0,6976	0,9110	0,5490	0,6976	0,9110	**0,6667**	**0,7668**	**0,9206**
Pima	0,6153	0,7110	0,6707	0,5522	0,7170	0,5888	0,6250	0,7120	0,6845	**0,6405**	**0,7355**	**0,6874**

Table 4 Result of the oversampling methods on all used datasets using KNN

Datasets	SMOTE			K-means SMOTE			ADASYN			DP-SMOTE		
	F-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M
Pima	0,5890	0,6803	0,6568	0,5352	0,6796	0,5861	0,5714	0,6470	0,6568	**0,6289**	**0,7165**	**0,6865**
Vehicle1	0,5542	0,6687	0,7310	0,5313	0,6815	0,6525	0,5506	0,6656	0,7418	**0,5750**	**0,6839**	**0,7431**
Ecoli2	0,8000	0,8438	0,9320	0,8750	0,9058	0,9468	0,5957	0,7115	0,8705	**0,9032**	**0,9316**	**0,9516**
Ecoli4	0,7999	0,8581	0,9240	0,7999	0,8581	0,9240	0,7999	0,8696	0,9196	**0,8421**	**0,8945**	**0,9287**
Yeast3	0,6891	0,7741	**0,9041**	0,7723	0,8456	0,8967	0,6721	0,7630	0,9008	**0,8298**	**0,8975**	**0,9041**
Yeast4	0,2414	0,5695	0,7574	0,3590	0,6202	**0,7750**	0,2059	0,5563	0,7479	**0,3846**	**0,6597**	0,6620
Yeast5	0,7143	0,7872	0,9551	0,7273	0,7857	0,9864	0,7273	0,0,7857	**0,9865**	**0,7317**	**0,7988**	0,9562
Pima	0,5890	0,6803	0,6568	0,5352	0,6796	0,5861	0,5714	0,6470	0,6568	**0,6289**	**0,7165**	**0,6865**

Table 5 Result of the oversampling methods on all used datasets using GBDT

Datasets	SMOTE			K-means SMOTE			ADASYN			DP-SMOTE		
	F-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M	*F*-M	*AUC*	*G*-M
Pima	0,5811	0,7005	0,6310	0,5899	**0,7306**	0,6252	0,5935	0,6962	**0,6507**	**0,6069**	0,7273	0,6477
Vehicle1	0,6412	0,7484	0,7491	0,6230	0,7519	0,7142	0,6316	0,7395	0,7455	**0,6615**	**0,7629**	**0,7634**
Ecoli2	0,7586	0,8699	0,8261	0,8148	**0,9113**	0,8349	0,7059	0,7974	0,8490	**0,8276**	**0,9113**	**0,8720**
Ecoli4	0,7500	0,9126	0,8002	0,8000	0,9842	0,8043	0,7059	0,8589	0,7960	**0,8750**	**0,9894**	**0,8731**
Yeast3	0,8081	0,8697	0,9124	0,8163	0,8770	0,9135	0,8077	0,8569	0,9339	**0,8485**	**0,8911**	**0,9392**
Yeast4	0,2778	0,5929	0,6543	0,3226	0,6180	0,6581	0,2564	0,5821	0,6519	**0,3636**	**0,6738**	0,5935
Yeast5	0,6667	0,8177	0,8187	0,6250	0,8055	0,7799	0,6250	0,8055	0,7797	**0,6875**	**0,8379**	**0,8198**
Pima	0,5811	0,7005	0,6310	0,5899	**0,7306**	0,6252	0,5935	0,6962	**0,6507**	**0,6069**	0,7273	0,6477

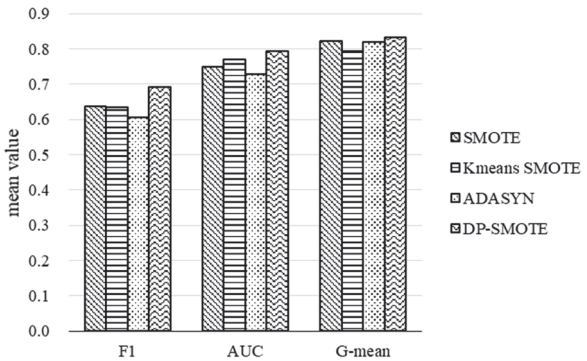


Figure 3 Mean values of all methods on all tested datasets for SVM

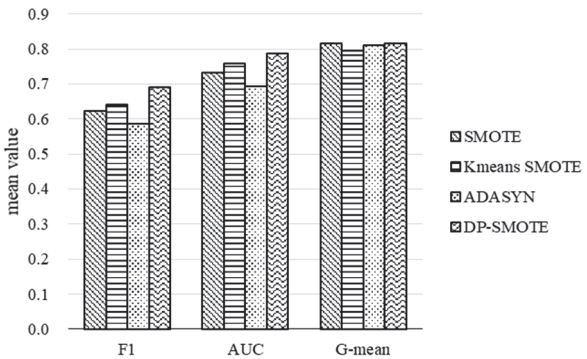


Figure 4 Mean values of all methods on all tested datasets for KNN

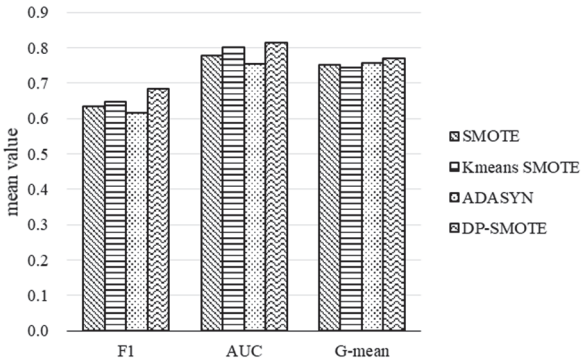


Figure 5 Mean values of all methods on all tested datasets for GBDT

4.5 Parameters Analysis of the Proposed Method

Since the proposed method adopts density peaks clustering to classify sub-clusters in minority class, the selection of the corresponding parameters for density peaks Clustering is crucial. Since density peaks clustering does not require any initial cluster centers and number of clusters as prior knowledge, the cut-off distance d_c is key to its clustering performance.

Therefore, we let the ratio of included samples on average be t, and then we changed the ratio t from 1% to 5% to investigate the effect of the cut-off distance d_c on the classification performance of our proposed method. Only the KNN, which was the better classifier in the comparison test, was used as the classifier on the KEEL datasets to observe the changes in the evaluation metrics *F*-Measure, *AUC* and *G*-mean. The results were averaged across the datasets after 10-fold cross-validation at the same value. The effects of different truncation distances on the evaluation metrics are shown in Fig. 6 to Fig. 8.

Fig. 6 to Fig. 8 show a general trend of increasing and then decreasing performance of the evaluation indicators. In the initial stage, the classification performance increases with increasing d_c value. When a certain value is reached, the classification performance remains stable and decreases with further increase of d_c. When d_c is taken as 2%, all three evaluation indicators can achieve better results.

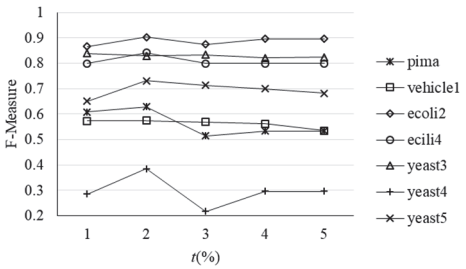


Figure 6 Effect of different d_c values on *F*-Measure

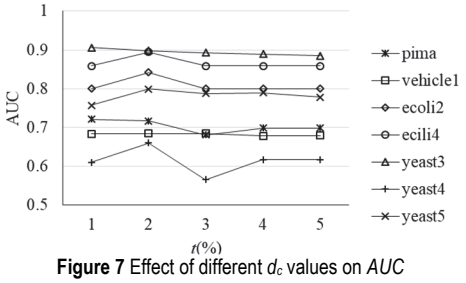


Figure 7 Effect of different d_c values on *AUC*

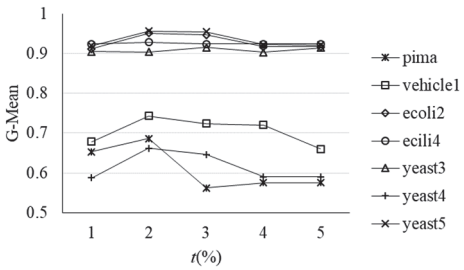


Figure 8 Effect of different d_c values on *G*-Mean

5 CONCLUSION

To address the problem of unbalanced data classification, in this paper a data processing method based on density peaks clustering is proposed. Firstly, density peaks clustering is introduced in minority class to accurately and quickly identify sub-clusters, avoiding spatial shape as well as parameter restrictions. Secondly, the oversampling coefficient is adjusted according to the number of oversampling of each sub-cluster to ensure that smaller sub-clusters generate more instances and solve the intra-class imbalance of minority class. Thirdly, the SMOTE interpolation formula is improved to interpolate between cluster cores and other instances of the same sub-cluster to prevent the generated instances from falling into the majority class region and to reduce the generation of noise points and overlapping instances. Finally, comparative experiments are carried out on the artificial dataset and the KEEL datasets to verify that the proposed method effectively reduces the introduction of noise and overlapping instances and improves the classification performance of unbalanced data. In addition, the effect of truncation distance on the performance of the proposed method is analyzed, and optimal parameters are suggested. In this paper, we only study the oversampling of binary imbalance. In practical applications, the multi-class problem is more common, and the next step can be explored for the oversampling of multi-class imbalance.

6 REFERENCES

[1] Tao, X. M., Li, Q., Guo, W. J., Ren, C., Li, C. X., Liu, R., & Zou, J. R. (2019). Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. *Information Sciences*, *487*, 31-56. https://doi.org/10.1016/j.ins.2019.02.062

[2] Yuan, X. H., Xie, L. J., & Abouelenien, M. (2018). A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data. *Pattern Recognition*, *77*, 160-172. https://doi.org/10.1016/j.patcog.2017.12.017

[3] Han, S. Y., Choi, H. J., & Oh, J. S. (2019). Fault Diagnosis of Planetary Gear Carrier Packs: A Class Imbalance and Multiclass Classification Problem. *International Journal of Precision Engineering and Manufacturing*, *20*(2), 167-179. https://doi.org/10.1007/s12541-019-00082-4

[4] Fiore, U., Santis, A. D., Perla, F., & Palmieri, F. (2017). Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, *479*,448-455. https://doi.org/10.1016/j.ins.2017.12.030

[5] Tao, X., Li, Q., Guo, W., Ren, C., He, Q., Liu, R. & Zou, J. R. (2020). Adaptive weighted over-sampling for imbalanced datasets based on density peaks clustering with heuristic filtering. *Information Sciences*, *519*, 43-73.

https://doi.org/10.1016/j.ins.2020.01.032
[6] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Artificial Intelligence Research*, *16*, 321-357. https://doi.org/10.1613/jair.953
[7] He, H. B., Bai, Y., Garcia, E. A., & Li, S. T. (2008). Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, 1322-1328.
[8] Barua, S., Islam, M. M., & Murase, K. (2011). A novel synthetic minority oversampling technique for imbalanced data set learning. *Springer-Verlag*, 735-744. https://doi.org/10.1007/978-3-642-24958-7_85
[9] Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2012). Density-based synthetic minority over-sampling technique. *Applied Intelligence*, *36*(3), 664-684. https://doi.org/10.1007/s10489-011-0287-y
[10] Rodriguez, A. & Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, *344*(6191), 1492. https://doi.org/10.1126/science.1242072

Contact information:

Jie CAO
(Corresponding author)
Nanjing University of Information Science & Technology,
No. 219, Ningliu Road, Nanjing, Jiangsu, China,
Xuzhou University of Technology,
No. 2 Lishui Road, Xuzhou, Jiangsu, China
E-mail: 857415571@qq.com

Yong SHI
Nanjing University of Information Science & Technology,
School of Mathematics and Statistics,
No. 219, Ningliu Road, Nanjing, Jiangsu, China
E-mail: 1799706316@qq.com