Soft Computing (2022) 26:2781–2792
https://doi.org/10.1007/s00500-021-06654-9

DATA ANALYTICS AND MACHINE LEARNING

Binary imbalanced big data classification based on fuzzy data reduction and classifier fusion

Junhai Zhai[1] · **Mohan Wang**[1] · **Sufang Zhang**[2]

Accepted: 6 December 2021 / Published online: 28 January 2022

Abstract
The era of big data has arrived, making it impossible for traditional machine learning algorithms to perform training in a stand-alone computing environment. In this paper, we propose a method for imbalanced binary classification of large-scale datasets based on undersampling and ensemble. More specifically, our method first adaptively partitions the majority class big data into k clusters, followed by undersampling to create k balanced datasets. Subsequently, k base classifiers are trained on each balanced dataset and are combined to perform the final prediction. Existing undersampling methods randomly select a subset of the majority class; thus, important instances may be lost during the process. In contrast, our proposed fuzzy data reduction scheme selects informative instances from each cluster, preventing information loss. Traditional ensemble methods have negative correlations between the base classifiers, whereas our proposed classifier fusion scheme fuses the base classifiers using fuzzy integral to facilitate modeling the relations between the base classifiers. The proposed algorithm is evaluated on six imbalanced large data sets and compared with state-of-the-art undersampling and ensemble methods, including the synthetic minority oversampling technique bagging (SMOTE-Bagging), SMOTE-Boost, and Binary Ensemble Classification for Imbalanced big data based on MapReduce and Upper sampling (BECIMU). Quantitative evaluations and theoretical analysis demonstrate that the proposed method outperforms the three state-of-the-art methods by 1.47%, 2.00% and 2.03%, and by 3.15%, 2.15% and 2.52%, in terms of the average G-mean and AUC-area, respectively.

Keywords Big data · Big data platform · Imbalanced data classification · Fuzzy data deduction · Fuzzy integral

1 Introduction

Many real-life binary imbalanced big data classification problems exist, for example, extreme weather prediction (Wang and Ding 2015), software defect prediction (Zhong et al. 2016), machinery fault diagnosis (Ding et al. 2017), spam filtering (Murtaza et al. 2020), and medical image classification (Murtaza et al. 2020). Since the class imbalance problem was originally proposed by Japkowicz (2000), different researchers have developed many methods. However, most of them focus only on small-scale datasets. With the emergence of big data, it is impossible for conventional machine learning algorithms to perform training in a stand-alone computing environment.

In this paper, we propose an imbalanced binary classification method based on undersampling and ensemble method for large-scale datasets. Specifically, we propose to (1) adaptively partition the majority class of the big data into k clusters using the open-source big data platforms, (2) use undersampling to create k balanced datasets and (3) train k base classifiers on the balanced datasets which are combined to perform the final prediction.

Undersampling (Japkowicz 2000; Liu et al. 2009; Ofek et al. 2017; Lin et al. 2017) is a popular method for imbalanced binary classification. Let S be an imbalanced data set, $S = S^+ \cup S^-$, where S^+ and S^- denote the positive class (minority class) and the negative class (majority class). First, a subset S' from S^- is randomly selected to ensure that $|S'| = |S^+|$. Then, a balanced training set S_{tr} is obtained by combining S' and S^+. Finally, a classifier is trained on S_{tr} to classify the imbalanced test set S_{te}. Although the under-

✉ Junhai Zhai
mczjh@126.com

[1] Hebei Key Laboratory of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, No. 180 Wusi East Road, Baoding 071002, Hebei Province, China

[2] Hebei Branch of China Meteorological Administration Training Centre, China Meteorological Administration, 434 Yuhua East Road, Baoding 071000, Hebei Province, China

sampling method is straightforward, it has the drawback that some essential negative class instances may be lost due to random sampling. The ensemble method fuses the base classifiers (which are often weaker) to create a stronger classifier. It should be noted that the base classifiers are not independent because the k balanced training subsets contain the same positive class subset. Existing studies (Wang et al. 2009; Chawla et al. 2003b; Zhai et al. 2018a) failed to consider this, resulting in negative correlations between base classifiers.

To address these problems, we propose fuzzy data reduction and classifier fusion schemes. After the negative big data set S^- has been clustered into k clusters $S_1^-, S_2^-, \ldots, S_k^-$ using an adaptive clustering method[1], fuzzy data reduction is adopted to select the informative instances from S_i^- for each obtained cluster $S_i^-(1 \leq i \leq k)$. As a result, k undersampled negative class subsets $R_i^-(1 \leq i \leq k)$ are obtained. Selecting informative instances instead of random instances guarantees that less information is lost during the undersampling process. Given the k balanced training sets $S_i = R_i^- \cup S^+(1 \leq i \leq k)$, the base classifiers are trained and fused using fuzzy integral, which can accurately model the relations between the k base classifiers. The fusion via fuzzy integral can reduce the negative correlations between the base classifiers and enhance the classification accuracy of binary imbalanced data.

To summarize, we propose a binary imbalanced big data classification approach based on fuzzy data reduction and classifier fusion. The main contributions of this paper are threefold. *First*, we propose an undersampling method for negative big data based on fuzzy data reduction to minimize information loss. *Second*, we present a binary imbalanced big data classification approach based on classifier fusion to prevent negative correlations between base classifiers. *Third*, extensive experiments are conducted on two big data open-source platforms (MapReduce and Spark) to compare the G-mean and AUC-area of the proposed methods and three state-of-the-art methods on six binary imbalanced big data sets. In addition, we present a theoretical analysis on the running time, the number of task synchronizations, and the number of files of the proposed methods implemented on the MapReduce and Spark.

2 Related works

Existing binary imbalanced data classification methods can be classified into three categories: data-level, algorithm-level, and ensemble methods. Since this paper focuses on undersampling and ensemble methods, we only review these two methods.

[1] k is automatically determined by the clustering algorithm.

2.1 Undersampling methods

Undersampling is a popular method that uses a subset of the majority class to deal with class imbalance (Liu et al. 2009). Japkowicz and Stephen (2002) provided a theoretically proof that classifiers trained on the sample data set provided equivalent generalization performance to classifiers trained on the original data set. Ofek et al. (2017) proposed a clustering-based undersampling technique that clusters the minority class instances and selects a similar number of majority class instances from each cluster. The algorithm exhibited high predictive performance with linear complexity bound by the size of S^+. Bach et al. (2019) proposed a clustering-based undersampling method that clusters the majority class instances and removes the instances from the high-density domains in contrast to the approach in Ofek et al. (2017), which clusters the minority class instances. Lin et al. (2017) introduced two clustering-based undersampling strategies, in which set the number of clusters in the majority class is equal to the number of data points in the minority class. The Tomek-link (T-Link) was proposed as a data compression and cleaning technique in Tomek (1976). Given two instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belonging to different classes, $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ be the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. A pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is called a T-Link, if there is no instance $\boldsymbol{x}_l$, such that $d(\boldsymbol{x}_i, \boldsymbol{x}_l) < d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ or $d(\boldsymbol{x}_j, \boldsymbol{x}_l) < d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. If a T-Link exists between 2 instances, one of the instances is noise, or both are borderline instances. Batista et al. (2004) used the T-Link as an undersampling method; only majority instances were removed. Kang et al. (2017) proposed an undersampling scheme that incorporates a noise filter for minority samples before the undersampling step. In undersampling, the deleted samples are never used to train the classifier, which may result in information loss. Therefore, Fan et al. (2016) presented a one-sided dynamic undersampling (ODU) method that utilizes all samples for training and dynamically determines whether a majority sample should be used for classifier learning. The novelty of ODU is the dynamic undersampling of the majority class to balance the dataset. Vuttipittayamongkol and Elyan (2020) proposed an undersampling framework designed to identify and eliminate majority class instances from the overlapping region. Accurate identification and elimination of these instances maximizes the visibility of the minority class instances and prevents excess data reduction, minimizing information loss. Koziarski (2020) developed an undersampling method based on radial basis function for imbalanced data classification. García and Herrera (2009) were among the first researchers to investigate evolutionary undersampling (EUS). The objective of EUS is to increase the accuracy of the classifier by reducing instances mainly belonging to the majority class. A good trade-off is achieved between data reduction, data balancing, and classification accuracy by designing a suit-

able fitness function. The authors (García and Herrera 2009) proposed eight EUS methods and categorized them depending on the objective, selection scheme, and performance metrics. Triguero et al. (2015, 2016, 2017) extended the EUS approach to big data scenarios and proposed three EUS approaches for imbalanced big data classification. The first method (Triguero et al. 2015) is a divide-and-conquer approach based on the MapReduce paradigm. The drawback of this approach is the low density of the minority class in the subsets in extremely imbalanced cases. Reference Triguero et al. (2016) proposed the second approach to overcome this drawback; it was implemented on the Spark platform. The methods in Triguero et al. (2015) and Triguero et al. (2016) utilized divide-and-conquer strategy to split big data set into several subsets that are addressed individually. However, the global view of the data may be lost, reducing the model accuracy. The third method, a global EUS approach for imbalanced big data, addresses this problem (Triguero et al. 2017). Liang et al. (2021) proposed fast and efficient undersampling method for imbalanced learning. It combines the classification boundary adjustment and sample selection to improve the efficiency and effect of imbalanced learning. Zheng et al. (2021) proposed a three-stage undersampling method, in which noise removal, clustering and representative sample selection were carried out in the three stages, respectively. The method can overcome the shortcomings of undersampling methods based on clustering.

2.2 Ensemble methods

Ensemble methods used for the classification of imbalanced data can be divided into methods combined with undersampling and methods combined with oversampling approaches.

Regarding the first category, Liu et al. (2009) proposed two undersampling-based ensemble methods called EasyEnsemble and BalanceCascade for class imbalance learning. EasyEnsemble randomly samples l subsets from the majority class as training sets to train l base classifiers, l is a predefined parameter, and combines the outputs of the l classifiers. BalanceCascade trains the classifiers sequentially; the majority class instances that are correctly classified by the trained learners are removed in each step. Seiffert et al. (2010) proposed a simpler and faster ensemble method (RUSBoost) that combines random undersampling and a boosting algorithm. Galar et al. (2013) proposed EUSBoost, an improvement of the RUSBoost algorithm, which combines random undersampling with a boosting algorithm. EUSBoost has higher performance of the base classifiers than RUSBoost due to EUS approach. Besides, EUSBoost is more versatile because it uses different subsets of majority class instances to train each base classifier. Similarly, Sun et al. (2018) combined EUS with the bagging algorithm and proposed an EUS-based ensemble method called EUS-Bag. The advantage of EUS-Bag is a new fitness function that considers performance, balance, and diversity. Lu et al. (2017) proposed a hybrid ensemble method that combines ensemble learning, undersampling techniques, and an adaptive boundary decision strategy. Sun et al. (2015) proposed a split balancing ensemble (SBE) approach for solving the class-imbalance problem. The SBE randomly partitions the set of majority class into several subsets with same sizes as the set of minority class. Each subset is combined with the minority class instances to obtain balanced subsets. The basic classifiers of the ensemble approach are trained on these balanced subsets, and the outputs of the base classifiers are integrated using a combination rule. However, underfitting may occur if a training set with a high imbalance ratio is used in the SBE method. Chen et al. (2019) proposed a distance-based balanced ensemble (DBE) method for classifying data with a high imbalance ratio to handle this issue. The DBE divides the highly imbalanced data set into multiple imbalanced subsets with a much lower imbalance ratio and uses a modified adaptive semi-unsupervised weighted oversampling method for each subset to obtain balanced subsets to train base classifiers used in the ensemble approach. Guo et al. (2020) proposed a two-step ensemble learning method for classifying imbalanced data. In the first step, a projection matrix is used to enhance the separability between the diverse class examples to improve the performance of the base classifier. In the second step, undersampling is applied to improve the performance of the base classifiers in the minority class and further increase the diversity between the individual base classifiers. Wang et al. (2020c) proposed an entropy and confidence-based undersampling boosting framework called ECUBoost for imbalanced data sets. The entropy and confidence levels are used in ECUBoost to avoid losing informative samples, and ensure the validity and adequate structural distribution of the majority samples during undersampling. Yang et al. (2020) presented a hybrid classifier ensemble method for classifying imbalanced data. This approach combines density-based undersampling and cost-effective methods using state-of-the-art solutions and a multi-objective optimization algorithm. Raghuwanshi and Shukla (2019) proposed an undersampling-based ensemble method that creates several balanced training subsets by random undersampling of the majority class samples. The number of training subsets is determined by the degree of the class imbalance. The generated balanced training subsets are used for training the base classifiers, and bagging is used as the ensemble method. The drawback of this method is that the number of training subsets is very large if the original data set has a high imbalance ratio.

Regarding the second category, Chen et al. (2018) proposed an ensemble method for classifying imbalanced data. The method consists of two steps. First, it generates synthetic samples in a local domain of the training samples

and trains the base classifiers using the original training samples and synthetic neighborhood samples. Finally, the classifiers are fused for classifying imbalanced data. The proposed method addresses the class imbalance problem and promotes diversity. Chawla et al. (2003a) combined base the SMOTE algorithm and a boosting and proposed SMOTEBoost to improve prediction the accuracy of the minority class instances during boosting. Lim et al. (2017) developed an evolutionary cluster-based oversampling ensemble method (ECO-Ensemble) that combines a cluster-based synthetic data generation method with an evolutionary algorithm. Zhai et al. (2018b) presented an imbalanced big data classification algorithm that combines an oversampling method and an ensemble approach. Oversampling is carried out in an enemy nearest neighbor hypersphere of a positive instance, and the ensemble technique is implemented using fuzzy integral. The enemy nearest neighbor of each positive instance is obtained by Ren et al. (2017) proposed an ensemble-based adaptive over-sampling method for imbalanced data classification and used it for computer-aided detection of microaneurysm. Li et al. (2017) utilized the Wiener process oversampling (WPO) technique for classifying imbalanced data and combined it with ensemble learning to create the WPOBoost algorithm. Abdi and Hashemi (2015) integrated the Mahalanobis distance-based over-sampling (MDO) technique with a boosting algorithm and proposed the MDOBoost algorithm for multi-class imbalanced data. Galar et al. (2012) conducted a comprehensive review of ensemble methods for classifying imbalanced data, focusing on bagging and boosting. Huang et al. (2020) proposed an ensemble method based on conditional image generation (Zhai et al. 2021, 2019; Zhang et al. 2020) for imbalanced image classification, which uses the generative adversarial network for oversampling and uses data cleaning for downsampling. Yan et al. (2019) proposed an ensemble method based on the three-way decision model for imbalanced data classification. The key point of this method lie in considering the difference in the cost of selecting key samples selected by the three-way decision model.

The literature indicates a lack of studies on imbalanced big data classification, a topic that was only researched by Triguero et al. (2015, 2016, 2017). In this paper, we present a classification algorithm for binary imbalanced big data that combines fuzzy data reduction and classifier fusion. In the following section, we present the details of the proposed algorithm.

3 The proposed algorithm

In this section, we present the proposed algorithm in detail. Let $S = S^- \cup S^+$ be an imbalanced big data set, where S^- is an imbalanced big data set, and S^+ is a small or medium-size data set. The proposed algorithm is illustrated in Fig. 1. It consists of four stages: (1) Adaptively clustering negative big data; (2) calculating the reduction in each cluster; (3) constructing balanced training sets and training base classifiers; (4) integrating the trained base classifiers using fuzzy integral. We present the details of each stage below.

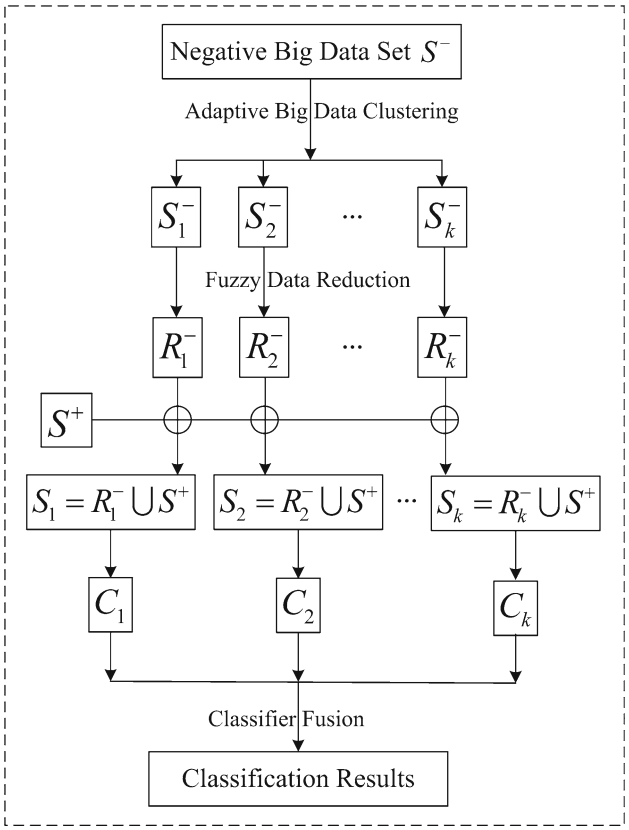

Fig. 1 The idea of the proposed algorithm

3.1 Adaptively clustering negative big data

The K-means algorithm is a very popular clustering algorithm; however, its major drawback is that the parameter K must be determined by the user. The X-means algorithm (Pelleg and Moore 2000) proposed by Pelleg and Moore overcomes this drawback. It is a hierarchical clustering approach that can efficiently estimate the parameter K by optimizing the Bayes information criterion (BIC). The X-means algorithm assumes a minimum number of clusters and dynamically increases them. It uses the BIC to guide splitting of clusters. If a single cluster (parent) is split into two clusters (children), and the BIC increases, two clusters are preferred to one cluster. Let $C_i (i = 1, 2)$ be the two child clusters; it is assumed that the data $\boldsymbol{x}$ contained in C_i follow

a d-dimensional normal distribution:

$$f(\boldsymbol{\theta}_i; \boldsymbol{x}) = (2\pi)^{-\frac{p}{2}} |\boldsymbol{\Sigma}_i|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}} \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)\right] \tag{1}$$

The calculation of BIC is given by equation (2).

$$\mathrm{BIC} = -2 \log L(\hat{\boldsymbol{\theta}}_i; \boldsymbol{x}) + q \log n_i \tag{2}$$

where $\hat{\boldsymbol{\theta}}_i = (\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)$ is the maximum likelihood estimate of the d-dimensional normal distribution; $\boldsymbol{\mu}_i$ is the d-dimensional means vector, and $\boldsymbol{\Sigma}_i$ is the $d \times d$ dimensional variance-covariance matrix; q is the number of the parameters. $\boldsymbol{x}$ is the d-dimensional data point in C_i; n_i is the number of elements in C_i. L is the likelihood function.

In this paper, we extend the X-means algorithm to the big data scenario and use it to cluster negative big data adaptively. The pseudocode of the extended X-means algorithm for big data is given in Algorithm 1.

Algorithm 1: Big data X-means algorithm

Input: The negative big data S^-; the minimum number of clusters K_{min}; the maximum number of clusters K_{max}.
Output: The adaptive clustering results: $S_1^-, S_2^-, \ldots, S_k^-$, the k is the optimal number of clusters.

```
Let K = K_min, use the big data K-means algorithm to S^-, and obtain K_min clusters;
for (each cluster in parentClusters) do
    if (K < K_max) then
        Calculate the parentBIC of each parent cluster by (2);
        Use the big data K-means algorithm to each parent cluster, and obtain two children clusters;
        Calculate the childBIC of each child cluster by (2);
        if (childBIC > parentBIC) then
            Add two children clusters to parentClusters;
        end
        else
            Output parentClusters to HDFS;
            Update K;
        end
    end
    else
        The remaining instances of the parent cluster form a cluster;
        Output parentClusters to HDFS;
    end
end
Output the adaptive clustering results: S_1^-, S_2^-, ..., S_k^-.
```

The main operation of the X-means algorithm is the big data K-means clustering, and the main computation is the calculation of the BIC. It is straightforward to compute a cluster's BIC due to the simplicity of estimating its mean vector $\hat{\boldsymbol{\mu}}_i$ and the covariance matrix $\hat{\boldsymbol{\Sigma}}_i$. Accordingly, the bottleneck of Algorithm 1 is the clustering of big data, which is performed using the big data computing framework MapReduce, as illustrated in Fig. 2.

Specifically, the process of big data K-means clustering based on MapReduce includes the following three stages:

(1) map: at each map node $i (1 \le i \le m)$, the distance between each sample $\boldsymbol{x}_{ij} \in S_i^- (1 \le i \le m; 1 \le j \le |S_i^-|)$ and each local cluster center $\boldsymbol{c}_{ik} \in C_{ik} (1 \le i \le m; 1 \le j \le K)$ is calculated in parallel, and $\boldsymbol{x}_{ij}$ is assigned to the nearest cluster.
(2) combiner: the local cluster center $\boldsymbol{c}_{ik} \in C_{ik}$ is updated in parallel by the formula (3),

$$\boldsymbol{c}_{ik} = \frac{1}{|C_{ik}|} \sum_{\boldsymbol{x}_{ij} \in C_{ik}} \boldsymbol{x}_{ik} \tag{3}$$

(3) At a reduce node, the global cluster center $\boldsymbol{c}_k \in C (1 \le k \le K)$ is updated by formula (4).

$$\boldsymbol{c}_k = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{c}_{ik} \tag{4}$$

3.2 Calculating the reduction in each cluster

After performing adaptively clustering, the negative big data set S^- is clustered into k subsets: $S_1^-, S_2^-, \ldots, S_k^-$. The negative class big data set is regarded as a k-class data set. We can use a data reduction approach (Wang et al. 2020a, 2019; Sun et al. 2019a; Ni et al. 2020, 2019) to eliminate unimportant data points from each cluster or use the instance selection method (Zhai et al. 2016; Wang et al. 2020b; Sun et al. 2019b) to select informative data points from each cluster. Since the cluster (or class) of a sample $\boldsymbol{x}$ in the local data subset S_i^- is known, data reduction or instance selection can be performed on a local data subset in parallel at each computing node.

In this paper, we use the fuzzy set method to calculate the reduction in $S_i^- (1 \le i \le k)$. Specifically, we calculate the reduction R_i^- of each S_i^- using the condensed fuzzy k-nearest neighbor (CFKNN) method. Why use this data reduction method because the k clusters are subsets of the negative big data set S^-, and they might overlap. CFKNN is an instance reduction or instance selection approach for fuzzy k-nearest neighbors (FKNN) (Keller et al. 2009) that overcomes the following three drawbacks of the k-nearest neighbor (KNN) method (Cover and Hart 1967).

(1) Given a test instance $\boldsymbol{x}$, the KNN method does not consider the difference in the contribution between the k nearest neighbors of $\boldsymbol{x}$ to classify $\boldsymbol{x}$.
(2) The KNN method does not consider the probability of $\boldsymbol{x}$ belonging to different classes.
(3) The KNN method is sensitive to noise.

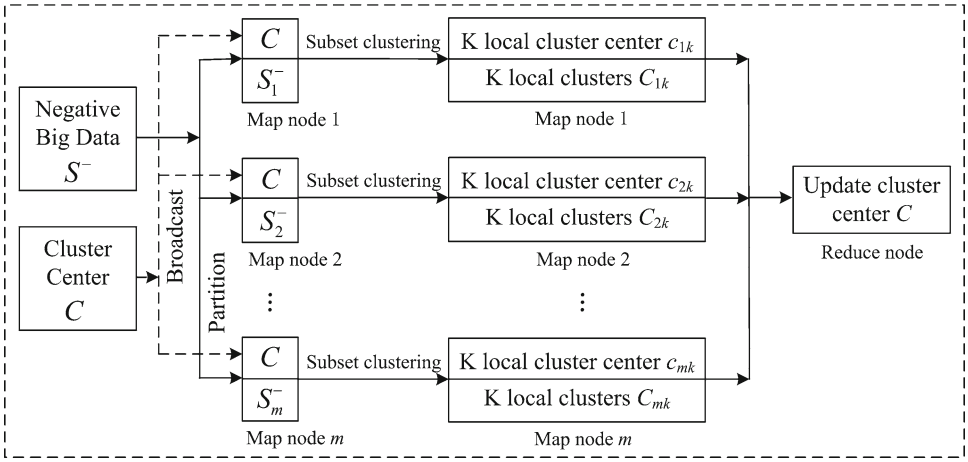

Fig. 2 The technical route of big data K-means clustering based on MapReduce

The FKNN method uses the fuzzy membership degree to describe the probability of $\boldsymbol{x}$ belonging to a class. The fuzzy membership degree of $\boldsymbol{x}$ is determined by its k nearest neighbors using Eq. (5).

$$\mu_j(\boldsymbol{x}) = \frac{\sum_{i=1}^{k} \mu_{ij} \left(\frac{1}{\|\boldsymbol{x}-\boldsymbol{x}_i\|^{\frac{2}{m-1}}} \right)}{\sum_{i=1}^{k} \left(\frac{1}{\|\boldsymbol{x}-\boldsymbol{x}_i\|^{\frac{2}{m-1}}} \right)} \tag{5}$$

where j is the index of classes, μ_{ij} is given by equation (6).

$$\mu_{ij} = \mu_j(\boldsymbol{x}_i) = \frac{\frac{1}{\|\boldsymbol{x}_i-\boldsymbol{c}_j\|^{\frac{2}{m-1}}}}{\sum_{j=1}^{k} \left(\frac{1}{\|\boldsymbol{x}_i-\boldsymbol{c}_j\|^{\frac{2}{m-1}}} \right)} \tag{6}$$

where $\boldsymbol{x}_i$ is the ith nearest neighbor of $\boldsymbol{x}$, $\boldsymbol{c}_j$ is the center of the jth class. In Eq. (5) and (6), m is a parameter that determines how the weight of the distance when calculating the neighbors' contributions to the membership value (Keller et al. 2009). In our experiments, we set $m = 2$, as suggested by Keller et al. (2009), i.e., the contribution of each neighboring point is weighted by the reciprocal of its distance from the point being classified.

In the CFKNN method, given an instance $\boldsymbol{x}$ in a subset $S_i^-(1 \le i \le k)$, we use the fuzzy membership degree $\mu_j(\boldsymbol{x})$ to calculate the information entropy $E(\boldsymbol{x})$ using Eq. (7).

$$E(\boldsymbol{x}) = -\sum_{i=1}^{k} \mu_j(\boldsymbol{x}) \log_2 \mu_j(\boldsymbol{x}) \tag{7}$$

The entropy is a measure of class uncertainty of the instances; the larger the entropy of an instance, the more difficult it is to determine its class. Accordingly, instances with larger information entropy are more informative. In the CKKNN method, we use entropy as a criterion to select informative instances. The pseudo-code of the CFKNN algorithm is given in Algorithm 2, where we omit the subscript of subset S_i^- for convenience; thus, S denotes the negative subset S_i^-.

Algorithm 2: The CFKNN algorithm

Input: An local negative subset S^-, and a threshold λ.
Output: A reduction R^- of S^-.

```
Randomly select k instances from each cluster to initialize R⁻,
and move the k instances from S⁻ to R⁻;
for (each x ∈ S⁻) do
    Find its k nearest neighbors in R⁻;
    for (each nearest neighbor of x) do
        Calculate its fuzzy membership degrees by Eq. (6);
    end
    Calculate the fuzzy membership degree of x by Eq. (5);
    Calculate the entropy of x by Eq. (7);
    if (E(x) > λ) then
        R⁻ = R⁻ ∪ {x};
    end
    return R⁻.
end
```

3.3 Constructing balanced training sets and training classifiers

In previous section, we obtained k reduction subsets, R_1, R_2, ..., R_k. Next, we construct k balanced training sets, S_1, S_2, ..., S_k, by unionizing each reduction subset S_i and the positive class subset S^+, i.e., $S_i = R_i^- \cup S^+ (1 \le i \le k)$. Next, we train k extreme learning machine (ELM) (Huang et al. 2006) classifiers $L_1, L_2, \ldots, L_k$, and their outputs are transformed into posterior probability by softmax function.

An ELM classifier is a Single hidden Layer Feed-forward neural Network (SLFN). A SLFN with m hidden nodes can be modeled with the following equation:

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} G(\boldsymbol{x}, \boldsymbol{w}_i, b_i)\boldsymbol{\beta}_i \tag{8}$$

where G denotes the hidden node activation function, $\boldsymbol{w}_i$ is the input weight vector connecting the i^{th} hidden node with the input nodes. b_i is the bias of the i^{th} hidden node. $\boldsymbol{\beta}_i$ is the output weight vector connecting the i^{th} hidden node with the output nodes. In ELM, $\boldsymbol{w}_i$ and b_i are randomly assigned, while $\boldsymbol{\beta}_i$ are analytically determined.

Given a training set, $S = \{(\boldsymbol{x}_i, y_i)|x_i \in R^d, y_i \in Y\}_{i=1}^n$, where $\boldsymbol{x}_i$ is an input vector and y_i is a class label in Y, $Y = \{\omega_1, \omega_2, \ldots, \omega_l\}$ be a set of class labels. Substitute $\boldsymbol{x}_i$ and y_i for x and $f(x)$ in (8), respectively, we obtain Eq. (9).

$$y_i = \sum_{j=1}^{m} G(\boldsymbol{x}_i, \boldsymbol{w}_j, b_j)\boldsymbol{\beta}_j \tag{9}$$

Eq. (9) can be written in a more compact format as

$$\boldsymbol{H}\boldsymbol{\beta} = \boldsymbol{Y} \tag{10}$$

where

$$\boldsymbol{H} = \begin{bmatrix} G(\boldsymbol{x}_1, \boldsymbol{w}_1, b_1) & \ldots & G(\boldsymbol{x}_1, \boldsymbol{w}_m, b_m) \\ \vdots & \ldots & \vdots \\ G(\boldsymbol{x}_n, \boldsymbol{w}_1, b_1) & \cdots & G(\boldsymbol{x}_n, \boldsymbol{w}_m, b_m) \end{bmatrix} \tag{11}$$

$$\boldsymbol{\beta} = (\boldsymbol{\beta}_1^T, \ldots, \boldsymbol{\beta}_m^T)^T \tag{12}$$

and

$$\boldsymbol{Y} = (\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_n^T)^T \tag{13}$$

Because usually the number of hidden nodes is much less than the number of training samples, $\boldsymbol{H}$ is a non-square matrix and one cannot expect an exact solution of the system (10). Yet, we can find its smallest norm least square solution by solving the optimization problem (14).

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{Y}\| \tag{14}$$

The smallest norm least-squares solution of (14) can be easily obtained using Eq. (15).

$$\hat{\boldsymbol{\beta}} = \boldsymbol{H}^{\dagger}\boldsymbol{Y} \tag{15}$$

where $\boldsymbol{H}^{\dagger} = \left(\boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{H}$ is the Moore–Penrose generalized inverse of matrix $\boldsymbol{H}$.

Given a test instance $\boldsymbol{x}$, the predicted posterior probability by softmax transformation is given using Eq. (16).

$$p(\omega_i|\boldsymbol{x}) = \frac{e^{y_i}}{\sum_{j=1}^{l} e^{y_j}} \tag{16}$$

3.4 Integrating the trained classifiers by fuzzy integral

Let $L = \{L_1, L_2, \ldots, L_k\}$ be the set of k ELM classifiers trained on the k constructed balanced training sets, $Y = \{\omega_1, \omega_2, \ldots, \omega_l\}$ be the set of class labels of the training instances. For test instance $\boldsymbol{x}$, the output of classifier L_i is a l-dimensional vector denoted by

$$L_i(\boldsymbol{x}) = (p_{i1}(\boldsymbol{x}), p_{i2}(\boldsymbol{x}), \ldots, p_{il}(\boldsymbol{x})) \tag{17}$$

where $p_{ij}(\boldsymbol{x}) \in [0, 1](1 \le i \le k; 1 \le j \le l)$ denotes the support degree given by classifier L_i to the hypothesis that $\boldsymbol{x}$ comes from class ω_j, $\sum_{j=1}^{l} p_{ij}(\boldsymbol{x}) = 1$, $p_{ij}(\boldsymbol{x})$ is estimated by Eq. (16).

The following matrix is called decision matrix Abdallah et al. (2012) with respect to $\boldsymbol{x}$.

$$DM(\boldsymbol{x}) = \begin{bmatrix} p_{11}(\boldsymbol{x}) & \cdots & p_{1j}(\boldsymbol{x}) & \cdots & p_{1l}(\boldsymbol{x}) \\ \vdots & & \vdots & & \vdots \\ p_{i1}(\boldsymbol{x}) & \cdots & p_{ij}(\boldsymbol{x}) & \cdots & p_{il}(\boldsymbol{x}) \\ \vdots & & \vdots & & \vdots \\ p_{k1}(\boldsymbol{x}) & \cdots & p_{kj}(\boldsymbol{x}) & \cdots & p_{kl}(\boldsymbol{x}) \end{bmatrix} \tag{18}$$

where the ith row of the matrix are the support degrees that classifier L_i classify $\boldsymbol{x}$ into classes $\omega_1, \omega_2, \ldots, \omega_l$, the jth column of the matrix are the support degrees from classifiers $L_1, L_2, \ldots, L_k$ for class ω_j.

Let $P(L)$ be the power set of L, the fuzzy measure on L is a set function $g : P(L) \to [0, 1]$, which satisfies the following two conditions:

(1) $g(\varnothing) = 1, g(L) = 1$;
(2) For $\forall A, B \subseteq L$, if $A \subset B$, then $g(A) \le g(B)$.

For $\forall A, B \subseteq L$ and $A \cap B = \varnothing$, g is called λ-fuzzy measure, if it satisfies the following condition:

$$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B) \tag{19}$$

where $\lambda > -1$ and $\lambda \neq 0$.

The value of λ can be obtained by solving the equation (20).

$$\lambda + 1 = \prod_{i=1}^{k}(1 + \lambda g_i) \tag{20}$$

where $g_i = g(\{L_i\})$, which is called fuzzy density of classifier L_i. It is noted that the equation (20) has only one solution which meets the conditions $\lambda > -1$ and $\lambda \neq 0$. Usually, g_i

can be determined using Eq. (21).

$$g_i = \frac{p_i}{\sum_{j=1}^{k} p_j}\delta. \tag{21}$$

where $\delta \in [0, 1]$ and p_i is testing accuracy or verification accuracy of classifier $L_i (1 \leq i \leq l)$.

Let $h : L \rightarrow [0, 1]$ be a function defined on L. The Choquet fuzzy integral (Abdallah et al. 2012) of function h with respect to g is defined using Eq. (22).

$$(C)\int h d\mu = \sum_{i=2}^{l+1}\left[h(L_{i-1}) - h(L_i)\right] g(A_{i-1}) \tag{22}$$

where $h(L_1) \geq h(L_2) \geq \cdots \geq h(L_k)$, $h(L_{l+1}) = 0$, $A_{i-1} = \{L_1, L_2, \ldots, L_{i-1}\}$.

Given a test instance $\boldsymbol{x}$, when we use fuzzy integral to integrate the k trained classifiers $L_1, L_2, \ldots, L_k$ for classifying $\boldsymbol{x}$, we first compute decision matrix $DM(\boldsymbol{x})$, and then sort jth$(1 \leq j \leq k)$ column of $DM(\boldsymbol{x})$ in descending order and obtain $(p_{i_1 j}, p_{i_2 j}, \ldots, p_{i_k j})$. The support degree $p_j(\boldsymbol{x})$ is calculated using Eq. (23).

$$p_j(\boldsymbol{x}) = \sum_{t=2}^{k+1}\left[p_{i_{t-1} j}(\boldsymbol{x}) - p_{i_t j}(\boldsymbol{x})\right] g(A_{t-1}) \tag{23}$$

The pseudo-code of integrating the trained classifiers by fuzzy integral is given in Algorithm 3.

Algorithm 3: The pseudo-code of integrating the trained classifiers by fuzzy integral

Input: $L = \{L_1, L_2, \cdots, L_k\}$ be a set of k base classifiers; $\boldsymbol{x}$ be a test instance.
Output: j^*, the class of $\boldsymbol{x}$.

```
for (i = 1; i ≤ k; i = i + 1) do
    Calculate the fuzzy densities g_i of classifier SLFN_i using Eq. (21);
end
Calculate parameter λ using Eq. (20);
Calculate DM(x) using Eq. (18);
for (j = 1; j ≤ l; j = j + 1) do
    Sort jth column of DM(x) in descending order and obtain (p_{i_1 j}, p_{i_2 j}, ··· , p_{i_k j});
    Set g(A_1) = g_{i_1};
    for (t = 2; t ≤ k; t = t + 1) do
        Calculate g(A_t) = g_{i_t} + g(A_{t-1}) + λ g_{i_t} g(A_{t-1});
    end
    Calculate p_j(x) = Σ_{t=2}^{k+1} [p_{i_{t-1} j}(x) − p_{i_t j}(x)] g(A_{t-1});
end
Calculate p_{j*}(x) = argmax_{1≤j≤l} {p_j(x)};
Output j*.
```

Table 1 Confusion matrix of binary imbalanced classification problem

True labels	Prediction labels	
	Yes	No
Positive	TP (True positive)	FN (False negative)
Negative	FP (False positive)	TN (True negative)

Table 2 The mean vectors and covariance matrices of Gaussian 1

i	$\boldsymbol{\mu}_i$	$\boldsymbol{\Sigma}_i$
1	$(1.0, 1.0)^{\mathrm{T}}$	$\begin{bmatrix} 0.6 & -0.2 \\ -0.2 & 0.6 \end{bmatrix}$
2	$(2.5, 2.5)^{\mathrm{T}}$	$\begin{bmatrix} 0.2 & -0.1 \\ -0.1 & 0.2 \end{bmatrix}$

4 Experimental results and analysis

We compared the proposed method with three state-of-the-art approaches on a big data platform with 8 computing nodes. The three approaches are SMOTE-Bagging (Wang et al. 2009), SMOTE-Boost (Chawla et al. 2003b), and BECIMU (Zhai et al. 2018a). The assessment metrics are G-mean and AUC-area which are commonly used for evaluating the performance of imbalanced data classification algorithms (Bach et al. 2019). The G-mean is defined in Eq. (24); it is obtained from the confusion matrix (contingency table) (Table 1). The AUC refers to the Area Under the Curve of receiver operating characteristics (ROC) (Liu et al. 2009).

$$\text{G-mean} = \sqrt{\frac{\text{TP}}{\text{TP+FN}} \times \frac{\text{TN}}{\text{TN+FP}}} \tag{24}$$

The data sets used in the experiments include 2 artificial data sets and 4 UCI data sets (Dua and Graff 2019). The first artificial data set (Gaussian 1) is a two-dimensional data set with two classes followed two Gaussian distributions whose mean vectors and covariance matrices are listed in Table 2. The second artificial data set (Gaussian 2) is a three-dimensional data set with four classes followed four Gaussian distributions whose the mean vectors and covariance matrices are listed in Table 3. The basic information of the 6 data sets is provided in Table 4, where #Negative and #Positive denote the number of negative and positive samples, respectively, and #Attribute denotes the number of attributes.

All experiments were conducted on a big data platform with 8 computing nodes; the configuration of the computing nodes is given in Table 5. It should be noted that the configuration of the master node and the slave node are the same in this platform.

We implemented the proposed algorithm using Hadoop and Spark on the big data platform. The G-means and AUC-

Table 3 The mean vectors and covariance matrices of Gaussian 2

i	$\boldsymbol{\mu}_i$	$\boldsymbol{\Sigma}_i$
1	$(0.0, 0.0, 0.0)^{\mathrm{T}}$	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
2	$(0.0, 1.0, 0.0)^{\mathrm{T}}$	$\begin{bmatrix} 1.0 & 0.0 & 1.0 \\ 0.0 & 2.0 & 2.0 \\ 1.0 & 2.0 & 5.0 \end{bmatrix}$
3	$(-1.0, 0.0, 1.0)^{\mathrm{T}}$	$\begin{bmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 6.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$
3	$(0.0, 0.5, 1.0)^{\mathrm{T}}$	$\begin{bmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 3.0 \end{bmatrix}$

Table 4 The basic information of the 6 data sets

Data sets	#Negative	#Positive	#Attribute
Gaussian 1	499,950	13,000	2
Gaussian 2	749,950	19,000	3
MiniBooNE	93,507	2200	50
Skin	194,148	4800	4
Healthy	70,216	1800	9
Hepmass	800,000	20,000	28

Table 5 The configuration of the nodes in the cloud computing platform

Items	Configuration
CPU	Inter Xeon E5-4603 with two cores, 2.0GZ
Memory	16GB
Network card	Broadcom 5720 QP 1Gb
Hard disk	1TB
Operating system	CentOS 6.4
Hadoop	Hadoop 2.7.1
Sprk	Spark 2.3.1
JDK	JDK 1.8

area of the proposed algorithm and the three state-of-the-art methods are listed in Tables 6 and 7, and Tables 8 and 9, respectively.

The results indicate that the proposed method achieved 5 maximum of G-mean (bold values in column 5 of Tables 6 and 7), SMOTE-Boost achieved another maximum of G-mean (bold values in column 3 of Tables 6 and 7). The experiment results of AUC-area are similar to those of G-mean (bold values in Tables 8 and 9). Overall, the proposed

Table 6 The experimental results of G-mean compared with the three state-of-the-art methods with Hadoop

Data sets	SMOTE-Bagging	SMOTE-Boost	BECIMU	Proposed method
Gaussian 1	0.9011	0.9106	0.9039	**0.9357**
Gaussian 2	0.8369	**0.8600**	0.8209	0.8216
MiniBooNE	0.8888	0.8907	0.8497	**0.8934**
Skin	0.8944	0.8606	0.8939	**0.9218**
Healthy	0.8850	0.8700	0.8764	**0.8999**
Hepmass	0.8755	0.8709	0.8915	**0.9015**

Table 7 The experimental results of G-mean compared with the three state-of-the-art methods with Spark

Data sets	SMOTE-bagging	SMOTE-boost	BECIMU	Proposed method
Gaussian 1	0.9034	0.9103	0.9017	**0.9413**
Gaussian 2	0.8436	**0.8574**	0.8436	0.8237
MiniBooNE	0.8897	0.8915	0.9027	**0.9055**
Skin	0.8894	0.8637	0.8946	**0.9227**
Healthy	0.8943	0.8658	0.8834	**0.9013**
Hepmass	0.8865	0.8738	0.8947	**0.8963**

Table 8 The experimental results of AUC-area compared with the three state-of-the-art methods with Hadoop

Data sets	SMOTE-bagging	SMOTE-boost	BECIMU	Proposed method
Gaussian 1	0.8834	0.9057	0.8933	**0.9254**
Gaussian 2	0.7999	**0.8618**	0.8017	0.8313
MiniBooNE	0.8739	0.8825	0.8604	**0.9010**
Skin	0.8891	0.8713	0.9008	**0.9175**
Healthy	0.8901	0.8890	0.8644	**0.9004**
Hepmass	0.8816	0.8734	0.8894	**0.9109**

Table 9 The experimental results of AUC-area compared with the three state-of-the-art methods with Spark

Data sets	SMOTE-bagging	SMOTE-boost	BECIMU	Proposed method
Gaussian 1	0.9088	0.9008	0.9205	**0.9510**
Gaussian 2	0.8390	**0.8666**	0.8325	0.8515
MiniBooNE	0.8739	0.8801	0.8966	**0.9138**
Skin	0.8916	0.8874	0.9017	**0.9300**
Healthy	0.8745	0.8900	0.8984	**0.9127**
Hepmass	0.8698	0.8877	0.8915	**0.9084**

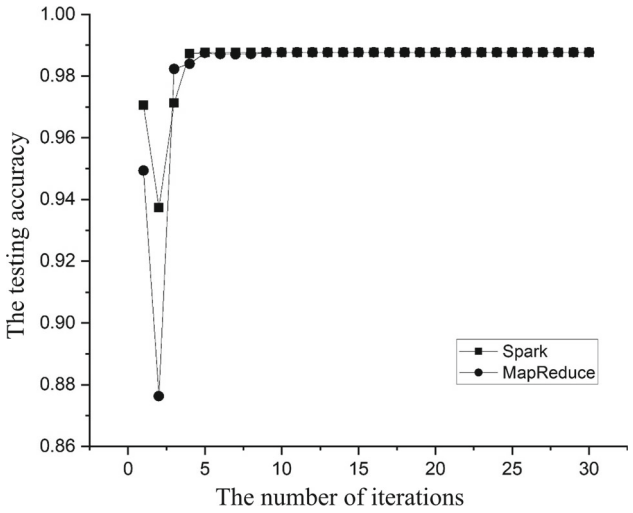

Fig. 3 The relationship between testing accuracy and number of iteration on Hadoop and Spark

method outperformed the 3 state-of-the-art methods. We believe that the proposed method is superior to the 3 state-of-the-art methods for the following three reasons:

(1) Adaptive clustering of the negative class big data partitions the data into several groups and maintains the intrinsic distribution.
(2) As a heuristic undersampling method, the instance selection prevents the loss of useful samples by random undersampling and selects informative samples from each cluster.
(3) Since the training sets used for training the base classifiers are not independent, they include the same positive subset. In other words, there are correlations between the base classifiers. The correlations can be positive, the base classifiers enhance each other in this case. The correlations can also be negative, the base classifiers restrain each other in this situation. The fuzzy integral can accurately model the two types of correlations between the base classifiers, increasing the classification performance of the ensemble learning system.

If an algorithm is implemented on different big data platforms, there should be no statistical difference in the testing accuracy. Figure 3 shows the experimental results on the Gaussian 1 data set on Hadoop and Spark. However, the number of files, number of task synchronizations, and running times may be significantly different for the two big data platforms. Therefore, we conducted a theoretical analysis regarding these three aspects.

The number of files refers to the number of intermediate files produced when the algorithm runs on the two big data platforms Hadoop and Spark. The number of intermediate files not only affect occupy the memory space but also affects the input/output (I/O) performance, potentially increasing the running time of the algorithm. On the Hadoop platform, the shuffle operation of MapReduce sorts and merges the intermediate results produced by the map task. MapReduce reduces the amount of data transferred between the computing nodes by merging and sorting the intermediate results. As a result, each map task produces only one intermediate data file. In contrast, the Spark platform does not have a merge and sort operation for intermediate data files, and data from different partitions are saved in a single file, i.e., the number of partitions is the number of intermediate files.

Regarding the number of task task synchronization, the reduce operation cannot be performed until all map operations are completed because MapReduce is a synchronous model. Spark is an asynchronous model, the number of synchronizations is larger on Hadoop than on Spark. The fewer synchronizations, the faster the algorithm is executed.

The running time T of the algorithm is determined by the sorting time T_{sort} and the transfer time T_{trans} of the intermediate data. When MapReduce sorts and merges the intermediate results, we assume that each map task requires m splits of the data, and each reduce task requires r splits of the data; thus, the sorting time of the intermediate data is $T_{\text{MR-sort}} = m \log m + r$. Since $r \leq m$ in most cases, $T_{\text{MR-sort}} = O(m \log m)$. In contrast, Spark has no shuffling process; therefore, $T_{\text{Sport-sort}} = 0$. T_{trans} is determined by the size of the intermediate data $|D|$ and the speed of network transmission C_r. If we ignore the difference between network transmission speeds, $T_{\text{trans}} \propto |D|$. The difference in trans-

Table 10 The comparison of file number, synchronous number and running time on Hadoop and Spark

Data sets	File number		Synchronous number		Running time/s	
	Hadoop	Spark	Hadoop	Spark	Hadoop	Spark
Gaussian 1	168	1400	29	19	6883	1101
Gaussian 2	168	1400	29	19	40391	6660
MiniBooNE	168	1400	29	19	8267	1759
Skin	168	1400	29	19	237	169
Healthy	252	3080	109	37	468	353
Hepmass	210	1600	47	28	1,018,871	18,189

mission time between Hadoop and Spark depends largely on the number of synchronizations. Spark uses pipeline technique to reduce the number of synchronizations, as the number of iterations increases, Spark has more advantages over MapReduce on T_{trans}. We summarize the results of the number of files, number of task synchronizations, and running time of the proposed method in Table 10. The results are consistent with the results of the above analysis.

5 Conclusion

A binary imbalanced classification method for big data based on fuzzy data reduction and classifier fusion via a fuzzy integral was proposed in this paper. The proposed method has three advantages: (1) It uses MapReduce to cluster negative big data adaptively into subsets to maintain the intrinsic distribution of the data. (2) Heuristic undersampling (i.e., the instance selection) prevents the loss of useful samples, especially for imbalanced big data sets. Furthermore, the heuristic undersampling method can select informative samples from negative subset. (3) The ensemble method that uses a fuzzy integral improves the classification accuracy. Future studies will investigate (1) extending the proposed method to classifying multi-class imbalanced big data classification set and (2) conducting experimental comparisons with additional methods using more imbalanced big data sets and various evaluation indices.

Acknowledgements This study was supported by the key R&D program of science and technology foundation of Hebei Province (19210310D), and by the natural science foundation of Hebei Province (F2021201020).

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abdallah ACB, Frigui H, Gader P (2012) Adaptive local fusion with fuzzy integrals. IEEE Trans Fuzzy Syst 20(5):849–864

Abdi L, Hashemi S (2015) To combat multi-class imbalanced problems by means of over-sampling and boosting techniques. Soft Comput 19:3369–3385

Bach M, Werner A, Palt M et al (2019) The proposal of undersampling method for learning from imbalanced datasets. Proc Comput Sci 159:125–134

Batista G, Prati R, Monard M (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newslett 6(1):20–29

Chawla NV, Lazarevic A, Hall LO et al (2003a) SMOTEBoost: Improving prediction of the minority class in boosting. Eur Conf Knowl Discov Databases 107–119

Chawla NV, Lazarevic A, Hall LO et al (2003b) SMOTEBoost: improving prediction of the minority class in boosting. Berlin, Heidelberg, European conference on principles of data mining and knowledge discovery. Springer, pp 107–119

Chen Z, Lin T, Xia X et al (2018) A synthetic neighborhood generation based ensemble learning for the imbalanced data classification. Appl Intell 48:2441–2457

Chen D, Wang XJ, Zhou CJ et al (2019) The distance-based balancing ensemble method for data with a high imbalance ratio. IEEE Access 7:68940–68956

Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inform Theory 13(1):21–27

Ding SF, Zhang N, Zhang J et al (2017) Unsupervised extreme learning machine with representational features. Int J Mach Learn Cybern 8(2):587–595

Dua D, Graff C (2019) UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA. http://archive.ics.uci.edu/ml

Fan Q, Wang Z, Gao DQ (2016) One-sided dynamic undersampling no-propagation neural networks for imbalance problem. Eng Appl Artif Intell 53:62–73

Galar M, Fernández A, Barrenechea E et al (2012) A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Trans Syst Man Cybern Part C Appl Rev 42(4):463–484

Galar M, Fernández A, Barrenechea E et al (2013) EUSBoost: enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. Patt Recogn 46:3460–3471

García S, Herrera F (2009) Evolutionary under-sampling for classification with imbalanced data sets: proposals and taxonomy. Evol Comput 17(3):275–306

Guo HP, Zhou J, Wu CA (2020) Ensemble learning via constraint projection and undersampling technique for class-imbalance problem. Soft Comput 24:4711–4727

Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501

Huang Y, Jin Y, Li Y et al (2020) Towards imbalanced image classification: a generative adversarial network ensemble learning method. IEEE Access 8:88399–88409

Japkowicz N (2000) The class imbalance problem: significance and strategies. In: Proceedings of the 2000 international conference on artificial intelligence, pp 111–117

Japkowicz N, Stephen S (2002) The class imbalance problem: a systematic study. Intell Data Anal 6(5):429–449

Kang Q, Chen XS, Li SS et al (2017) A noise-filtered undersampling scheme for imbalanced classification. IEEE Trans Cybern 47(12):4263–4274

Keller JR, Gray MR, Givens JA (2009) A fuzzy k-nearest neighbor algorithm. IEEE Trans Knowl Data Eng 21(9):1263–1284

Koziarski M (2020) Radial-based undersampling for imbalanced data classification. Patt Recogn 102:107262. https://doi.org/10.1016/j.patcog.2020.107262

Li Q, Li G, Niu W et al (2017) Boosting imbalanced data learning with Wiener process oversampling. Front Comput Sci 11:836–851

Liang T, Xu J, Zou B et al (2021) LDAMSS: Fast and efficient undersampling method for imbalanced learning. Appl Intell. https://doi.org/10.1007/s10489-021-02780-x

Lim P, Goh CK, Tan KC (2017) Evolutionary cluster-based synthetic oversampling ensemble (ECO-Ensemble) for imbalance learning. IEEE Trans Cybern 47(9):2850–2861

Lin WC, Tsai CF, Hu YH et al (2017) Clustering-based undersampling in class-imbalanced data. Inform Sci 409–410:17–26

Liu XY, Wu JX, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern Part B Cybern 39(2):539–550

Lu W, Li Z, Chu JH (2017) Adaptive ensemble undersampling-boost: a novel learning framework for imbalanced data. J Syst Softw 132:272–282

Murtaza G, Shuib L, Wahab AWA et al (2020) Deep learning-based breast cancer classification through medical imaging modalities: state of the art and research challenges. Artif Intell Rev 53:1655–1720

Ni P, Zhao SY, Wang XZ et al (2019) PARA: A positive-region based attribute reduction accelerator. Inform Sci 503:533–550

Ni P, Zhao SY, Wang XZ et al (2020) Incremental feature selection based on fuzzy rough sets. Inform Sci 536:185–204

Ofek N, Rokach L, Stern R et al (2017) Fast-CBUS: A fast clustering-based undersampling method for addressing the class imbalance problem. Neurocomputing 243:88–102

Pelleg D, Moore A (2000) X-means: extending K-means with efficient estimation of the number of clusters. In: Proceedings of the seventeenth international conference on machine learning (ICML 2000), pp 1–8

Raghuwanshi BS, Shukla S (2019) Class imbalance learning using underbagging based kernelized extreme learning machine. Neurocomputing 329:172–187

Ren FL, Cao P, Li W et al (2017) Ensemble based adaptive over-sampling method for imbalanced data learning in computer aided detection of microaneurysm. Comput Med Imag Graph 55:54–67

Seiffert C, Khoshgoftaar TM, Hulse JV et al (2010) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern Part A Syst Humans 40(1):185–197

Sun Z, Song Q, Zhu X et al (2015) A novel ensemble method for classifying imbalanced data. Patt Recogn 48(5):1623–1637

Sun B, Chen H, Wang JD et al (2018) Evolutionary under-sampling based bagging ensemble method for imbalanced data classification. Front Comput Sci 12:331–350

Sun L, Zhang XY, Qian YH et al (2019a) Feature selection using neighborhood entropy-based uncertainty measures for gene expression data classification. Inform Sci 502:18–41

Sun L, Zhang XY, Qian YH et al (2019b) Joint neighborhood entropy-based gene selection method with fisher score for tumor classification. Appl Intell 49(4):1245–1259

Tomek I (1976) Two modifications of CNN. IEEE Trans Syst Man Commun SMC 6:769–772

Triguero I, Galar M, Vluymans S et al (2015) Evolutionary undersampling for imbalanced big data classification. In: IEEE congress on evolutionary computation (CEC), 25–28 May 2015. Sendai, Japan, pp 715–722

Triguero I, Galar M, Merino D et al (2016) Evolutionary undersampling for extremely imbalanced big data classification under Apache Spark. In: IEEE congress on evolutionary computation (CEC), 24–29 July 2016. Vancouver, BC, Canada, pp 640–647

Triguero I, Galar M, Bustince H et al (2017) A first attempt on global evolutionary undersampling for imbalanced big data. In: IEEE congress on evolutionary computation (CEC), 5–8 June 2017. San Sebastian, Spain, pp 2054–2061

Vuttipittayamongkol P, Elyan E (2020) Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. Inform Sci 509:47–70

Wang DW, Ding W (2015) A hierarchical pattern learning framework for forecasting extreme weather events. In: 2015 IEEE international conference on data mining, 14–17 Nov, Atlantic City, NJ, USA, pp 1021–1025

Wang S, Yao X (2009) Diversity analysis on imbalanced data sets by using ensemble models. In: IEEE symposium on computational intelligence and data mining. Nashville, TN, USA, pp 324–331

Wang CZ, Huang Y, Shao MW et al (2019) Fuzzy rough set-based attribute reduction using distance measures. Knowl Based Syst 164:205–212

Wang CZ, Wang Y, Shao MW et al (2020a) Fuzzy rough attribute reduction for categorical data. IEEE Trans Fuzzy Syst 28(5):818–830

Wang CZ, Huang Y, Shao MW et al (2020b) Feature selection based on neighborhood self-information. IEEE Trans Cybern 50(9):4031–4042

Wang Z, Cao C, Zhu Y (2020c) Entropy and confidence-based undersampling boosting random forests for imbalanced problems. IEEE Trans Neural Netw Learn Syst. https://doi.org/10.1109/TNNLS.2020.2964585

Yan YT, Wu ZB, Du XQ et al (2019) A three-way decision ensemble method for imbalanced data oversampling. Int J Approx Reason 107:1–16

Zhai JH, Wang XZ, Pang XH (2016) Voting-based instance selection from large data sets with MapReduce and random weight networks. Inform Sci 367:1066–1077

Zhai JH, Zhang MY, Chen CX et al (2018a) Binary ensemble classification for imbalanced big data based on MapReduce and upper sampling. J Data Acquis Process 33(3):416–425 (**in Chinese**)

Zhai JH, Zhang SF, Zhang MY et al (2018b) Fuzzy integral-based ELM ensemble for imbalanced big data classification. Soft Comput 22(11):3519–3531

Zhai M, Chen L, Tung F et al (2019) Lifelong GAN: Continual learning for conditional image generation. IEEE/CVF Int Conf Comput Vis (ICCV) 2019:2759–2768. https://doi.org/10.1109/ICCV.2019.00285

Yang K, Yu Z, Wen X et al (2020) Hybrid classifier ensemble for imbalanced data. IEEE Trans Neural Netw Learn Syst 31(4):1387–1400

Zhai M. Y., Chen L, Mori G (2021) Hyper-LifelongGAN: scalable lifelong learning for image conditioned generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR2021), pp 2246–2255

Zhang M, Li T, Zhu R et al (2020) Conditional Wasserstein generative adversarial network-gradient penalty-based approach to alleviating imbalanced data classification. Inform Sci 512:1009–1023

Zheng M, Li T, Zheng X et al (2021) UFFDFR: Undersampling framework with denoising, fuzzy c-means clustering, and representative sample selection for imbalanced data classification. Inform Sci 576:658–680

Zhong GQ, Wang LN, Ling X et al (2016) An overview on data representation learning: from traditional feature learning to recent deep learning. J Finance Data Sci 2(4):265–278