Knowledge-Based Systems 136 (2017) 187–199

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Online feature selection for high-dimensional class-imbalanced data

Peng Zhou[a], Xuegang Hu[a,*], Peipei Li[a,*], Xindong Wu[b,*]

[a] Hefei University of Technology, Hefei 230009, China
[b] University of Louisiana, Lafayette, LA 70504, USA

ARTICLE INFO

Article history:
Received 9 February 2017
Revised 5 September 2017
Accepted 6 September 2017
Available online 8 September 2017

Keywords:
Online feature selection
High dimensional
Class imbalance
Neighborhood rough set

ABSTRACT

When tackling high dimensionality in data mining, online feature selection which deals with features flowing in one by one over time, presents more advantages than traditional feature selection methods. However, in real-world applications, such as fraud detection and medical diagnosis, the data is high-dimensional and highly class imbalanced, namely there are many more instances of some classes than others. In such cases of class imbalance, existing online feature selection algorithms usually ignore the small classes which can be important in these applications. It is hence a challenge to learn from high-dimensional and class imbalanced data in an online manner. Motivated by this, we first formalize the problem of online streaming feature selection for class imbalanced data, and then present an efficient online feature selection framework regarding the dependency between condition features and decision classes. Meanwhile, we propose a new algorithm of Online Feature Selection based on the Dependency in K nearest neighbors, called K-OFSD. In terms of Neighborhood Rough Set theory, K-OFSD uses the information of nearest neighbors to select relevant features which can get higher separability between the majority class and the minority class. Finally, experimental studies on seven high-dimensional and class imbalanced data sets show that our algorithm can achieve better performance than traditional feature selection methods with the same numbers of features and state-of-the-art online streaming feature selection algorithms in an online manner.

1. Introduction

Feature selection aims to select a subset of features, which can be used to derive a mapping function from samples to classes that is "as good as possible" according to some criterion [1]. There are many representative algorithms for traditional feature selection, such as ReliefF [2], Fisher Score [3], MI(Mutual Information)[4], mRMR (minimal Redundancy and Maximal Relevance) [5], Laplacian score [6,7], LASSO (least absolute shrinkage and selection operator) [8] and so on [9]. With the increasing of the scale of data, traditional batch feature selection can not meet the efficiency demand any more. For instance, the Web Spam Corpus 2011, a collection which has approximately 330,000 spam web pages and 16,000,000 features (attributes) [10]. Besides, all aforementioned approaches assume that all candidate features are available before learning takes place. However, in many real-world applications, features are generated dynamically, and arrive one by one over time. For example, in image analysis [11], multiple descriptors are extracted dynamically to capture various visual information of images, including HOG (Histogram of Oriented Gradients), color histogram and SIFT (Scale Invariant Feature Transform). Each of these descriptors is generated independently and image features are often expensive to generate and store and therefore may exist in a streaming format. Another example is the Mars crater detection from high resolution planetary images [12]. Tens of thousands of texture-based features, in different scales and different resolutions, can potentially be generated for high resolution planetary images. It is infeasible to acquire the entire feature set which means to have a near global coverage of the Martian surface. In order to deal with this challenge, many online streaming feature selection methods have been proposed [13].

Online feature selection with streaming features has attracted much attention in recent years and played a critical role in dealing with extremely high-dimensional problems [14–18]. Streaming features are defined as features that flow in one by one over time whereas the number of training examples remains fixed [15]. More specifically, Perkins and Theiler [19] considered the problem of online feature selection and proposed the Grafting algorithm based on a stagewise gradient descent approach. Grafting treats feature selection as an integral part of learning a predictor within a regularized framework. Zhou et al. [20] proposed two algorithms of information-investing and alpha-investing, based on

* Corresponding authors.
E-mail addresses: doodzhou@hotmail.com (P. Zhou), jsjxhuxg@hfut.edu.cn (X. Hu), peipeili@hfut.edu.cn (P. Li), xwu@louisiana.edu (X. Wu).

http://dx.doi.org/10.1016/j.knosys.2017.09.006
0950-7051/

streamwise regression for online feature selection. Alpha-investing does not need a global model and it is one of the penalized likelihood ratio methods. Wu et al. [15] presented an online streaming feature selection framework with two algorithms called OSFS (Online Streaming Feature Selection) and fast-OSFS. OSFS contains two major steps, including online relevance analysis and online redundancy analysis. Yu et al. [18] proposed the SAOLA approach (a Scalable and Accurate Online feature selection Approach) for high dimensional data. SAOLA employs novel online pairwise comparison techniques and maintains a parsimonious model over time in an online manner. Eskandari et al. [21] proposed a Rough Set based method for online streaming feature selection. The proposed algorithm uses classical significance analysis concepts in Rough Set theory to control an unknown feature space in online streaming feature selection problems.

Meanwhile, in many real-world applications, such as fraud and intrusion detection, text classification and medical diagnosis, in addition to high dimensionality, class imbalance is also very common [22,23]. For example, the number of fraud users is obviously much lower than that of normal users. In fact, the ratio of the small to the large classes can be drastic such as 1 to 100, 1 to 1,000, or 1 to 100,000 [24,25]. All these online streaming feature selection approaches mentioned above were proposed to deal with data sets with normal class distributions, thus, they cannot handle the class imbalance data effectively. It is hence a challenge for existing online streaming feature selection approaches.

To handle the issue of class imbalance in data sets, existing solutions mainly focus on two levels: the data level and the algorithmic level [26]. The former includes many different forms of re-sampling and the latter involves adjusting the costs of various classes, the probabilistic estimation and the decision threshold. In recent years, one class learning and feature selection for class imbalance data have also attracted much attention [26]. Feature selection can be very helpful for imbalanced data sets [27]. The aim of feature selection for imbalance data is to select features that can get higher separability between the majority class and the minority class. Existing works in feature selection for imbalance data are mostly batch algorithms [27–31]. For example, Zheng et al. [28] proposed a feature selection framework which selects positive features and negative features separately, and then explicitly combines them to improve the classification accuracy in the handling of class imbalance data. Hulse et al. [27] gave detailed comparisons of six commonly-used filters and three filters using classifier performance metrics on high-dimensional imbalance data. The biggest finding from this paper is that feature selection is beneficial to handle most high-dimensional imbalanced data sets. Wasikowski et al. [30] presented a first systematic comparison of three types of methods developed for imbalanced data classification problems (re-sampling, new algorithms and feature selection) and of seven feature selection metrics evaluated on small sample data sets from different applications. Results showed that the signal-to-noise correlation coefficient (S2N) and Feature Assessment by Sliding Thresholds (FAST) are great candidates for feature selection in most imbalanced applications, especially in the case of selecting a very small number of features. Maldonado et al. [29] proposed a backward elimination approach based on successive holdout steps, whose contribution measure is based on a balanced loss function obtained over an independent subset. Nevertheless, all aforementioned algorithms were proposed for traditional feature selection. To the best of our knowledge, there is no work relevant to online streaming feature selection for high-dimensional class imbalance data so far.

Rough set theory, proposed by Pawlak, has been proven to be an effective tool for feature selection, rule extraction and knowledge discovery [32]. Pawlak's rough sets were originally proposed to deal with categorical data. There are many works of using rough sets for attribute reduction and feature selection [33–36]. However, in real-world applications, there are many numerical features in data sets. Then, a neighborhood rough set that supports both continuous and discrete data was proposed to deal with this challenge [37]. There are some works using the neighborhood rough set for feature selection [38–42] and it has been proved as an effective approach in the handling of feature selection problems. We aim to apply neighborhood rough set theory in the handling of online streaming feature selection. This is because rough set based data mining does not require any domain knowledge [21]. In addition, we focus on class imbalance data where instances of the minority class are rare in data sets. We refine the neighborhood rough set method in this paper and use neighbors's class information for feature selection. The proposed neighborhood rough set based method does not need to consider the global class distribution of a data set which makes the impact of class imbalance be relatively small. However, all these neighborhood rough set based methods mentioned above were designed for traditional batch feature selection and there is no work of using a neighborhood rough set for feature selection in an online manner.

We would like to distinguish online streaming feature selection in this paper from previous studies of dynamic information systems in [43–47]. A complete information system is defined as $S = (U, C \cup D, V, f)$, where U is a non-empty finite set of objects, C is the set of condition attributes and D is the set of decision attributes. $V = \bigcup_{a \in A} V_a$, where V_a is a domain of attribute a and $A = C \cup D$. f: $U \times A \to V$ is an information function such that $f(x, a) \in V_a$ for every $x \in U$, $a \in A$. Nowadays, the dynamic information system learning approaches based on Rough Set theory mainly focus on the following two cases. (1) The object set in an information system evolves over time while the attribute set remains constant. (2) The attribute set in the information system evolves over time while the object set remains constant. Most existing efforts consider the situation where objects or features are available in the information system [48–51]. It is different from online streaming feature selection where the number of objects is fixed and the feature set grows with time. At each time stamp, we can just get one feature from the streaming features and the full feature space is unknown or inaccessible.

Motivated by this, in this paper, we first formalize the problem of online streaming feature selection for class imbalance data and then present an Online Feature Selection framework based on the dependency between the condition features and decision classes, named OFSD. Our contributions are as follows:

- We formally define the problem of online streaming feature selection for class imbalance data.
- We propose an online feature selection framework based on the dependency of either a single feature or a selected feature set to decision classes. To the best of our knowledge, all existing online feature selection methods measure features by a certain criterion individually and there is no related work considering the selected feature set as a group.
- A new Online Feature Selection algorithm based on the dependency between condition features and decision classes in K nearest neighbors (called K-OFSD) is proposed to handle the class imbalance data in an online manner. In order to select features which can get high separability between the small class and the large class, we make a full use of the neighbors' class information near to the target object. For the class imbalance problem, we refine the neighborhood rough set theory and use the information of a fixed number of nearest neighbors to select features, which can promote the dependency between condition features and decision classes.
- Extensive experimental studies show that our proposed algorithm can get better performance than traditional imbalanced

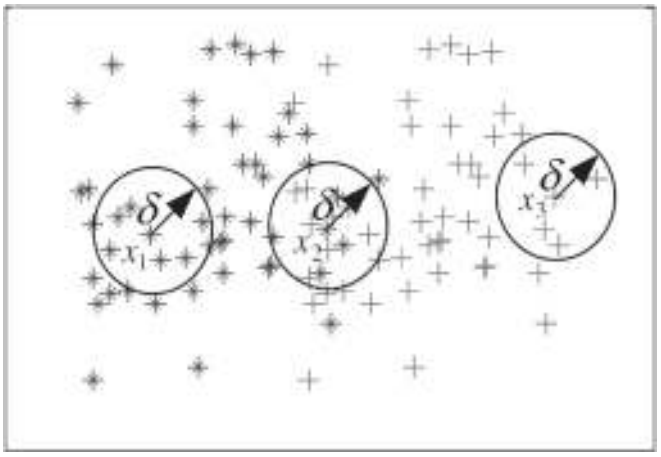

Fig. 1. δ neighborhood rough set.

feature selection approaches and state-of-the-art online streaming feature selection methods in the handling of high-dimensional class-imbalanced data.

The rest of the paper is organized as follows. Section 2 gives a brief introduction of neighborhood rough set theory. Section 3 presents our framework and the new algorithm. Section 4 reports experimental results and analyses all experimental algorithms. Section 5 concludes the paper.

2. Neighborhood rough set

Rough set theory, proposed by Pawlak, has been proven to be an effective tool for feature selection, rule extraction and knowledge discovery [32]. For classical rough set, the objects with the same feature values in terms of attributes B are drawn together and form an equivalence class denoted by $[x]_B$. The family of elemental granules $\{[x_i]_B, x_i \in U\}$ builds a concept system to describe an arbitrary subset of the sample space. Two unions of elemental granules are associated with X: lower approximation and upper approximation.

$$\underline{B}X = \{[x_i]_B \mid [x_i]_B \subseteq X\}; \tag{1}$$

$$\overline{B}X = \{[x_i]_B \mid [x_i]_B \cap X \neq \emptyset\}. \tag{2}$$

The lower approximation is the maximal union of elemental granules consistently contained in X, while the upper approximation is the minimal union of elemental granules containing X. The difference between lower approximation and upper approximation is called approximation boundary of $X : BN(X) = \overline{B}X - \underline{B}X$.

Pawlak's rough sets were originally proposed to deal with categorical data. However, in real-world applications, there are many numerical features in data sets. Then, a neighborhood rough set is proposed to deal with numerical data [37–40,52,53]. In simple terms, the neighborhood of x_i is a subset of samples close to x_i. There are some ways to define the neighborhoods of samples [39]. We can define it with a fixed radius from the prototype sample (δ neighborhood, as shown in Fig. 1 [52]) or the neighborhood with k samples (k-nearest neighborhood).

As shown in Fig. 1, all the δ neighbor samples of x_1 have the same class label C_1 of x_1 with mark "*" and the neighborhood samples of x_3 in a δ area are completely marked with "+" with class label C_2. Meanwhile, the samples in the neighborhood of x_2 come from classes C_1 and C_2 and we define the samples of x_2 are the boundary objects. In general, we need to find a feature subspace on which the boundary region is maintained as little as possible.

Table 1
An example dataset.

$x \in U$	f_1	f_2	f_3	f_4	d
x_1	3	5.6	-66	3.05	-1
x_2	5	6.9	95	4.84	1
x_3	8	5.3	-28	5.89	1
x_4	13	12.3	-35	6.14	1
x_5	6	15.2	72	6.55	-1
x_6	5	2.6	42	10.94	1
x_7	9	5.8	-33	23.85	-1
x_8	15	6.4	15	23.85	-1

For a class imbalanced data set, the number of instances of small class (C_{small}) is far less than the large class (C_{large}). If we use δ neighborhood information for feature selection, the instances of C_{small} can easily be overwhelmed by C_{large} and it is difficult to select the value δ. Thus, we use the k-nearest neighborhood relation for feature selection of class imbalance data. The definition of k-nearest neighborhood rough set is given below.

Definition 1. Considering object x and given a set of numerical attributes B to describe the object, we call the k-nearest-neighbors of x in terms of a k-nearest-neighbor information granule, denotes as $k_B(x_i)$.

$$k_B(x_i) = \{x_j \mid \Delta_B(x_i, x_j) \in Min_k\{Neighbors\}, x_j \in U\}, \tag{3}$$

where $Min_k\{Neighbors\}$ denotes the k nearest neighbors of x_i. Δ is a distance function (such as Euclidean distance), and $\Delta(x, y)$ denotes the distance between x and y. For $\forall x, y, z \in U$, it satisfies:

(1) $\Delta(x, y) \geq 0$, $\Delta(x, y) = 0$ if and only if $x = y$;
(2) $\Delta(x, y) = \Delta(y, x)$
(3) $\Delta(x, z) \leq \Delta(x, y) + \Delta(y, z)$

Table 1 shows an example dataset to be used to illustrate the definition of k-nearest-neighbors, where the distance function is calculated with Euclidean distance.

Let's take object x_3 and feature set $B = \{f_1, f_2\}$ with 2-nearest-neighbors as an example. First, we calculate all distances between x_3 and x_i $(i \neq 3)$ on B namely: $\Delta_B(x_3, x_1) = \sqrt{(8-3)^2 + (5.3-5.6)^2} = 5.009$, $\Delta_B(x_3, x_2) = 3.4$, $\Delta_B(x_3, x_4) = 8.602$, $\Delta_B(x_3, x_5) = 10.1$, $\Delta_B(x_3, x_6) = 4.036$, $\Delta_B(x_3, x_7) = 1.118$, $\Delta_B(x_3, x_8) = 7.086$. Then we can see that the first two smallest values are $\Delta(x_3, x_7)$ and $\Delta(x_3, x_2)$. Thus, the 2-nearest-neighbors of x_3 on feature set B are denoted as $k_B(x_3) = \{x_7, x_2\}$.

Like Pawlak's rough set model, we give the lower and upper approximations of k-nearest neighborhood as follows.

Definition 2. Given an arbitrary subset X of the sample space and a family of k-nearest-neighbor information granules $k_B(x_i)$, $i = 1, 2, \ldots, n$, we define the lower and upper approximations in terms of relation K as

$$\underline{K_B}X = \{x_i \mid k_B(x_i) \subseteq X, x_i \in U\}; \tag{4}$$

$$\overline{K_B}X = \{x_i \mid k_B(x_i) \cap X \neq \emptyset, x_i \in U\}. \tag{5}$$

The lower approximation of a decision, also called the positive region of the decision, is denoted by $POS_B(D)$.

Given a neighborhood decision system $NDT =< U, A, D >$, where U is the sample set, A is the condition attribute set and D is the decision attribute set. Assuming B is a subset of A, we have the definition of dependency degree as follows.

Definition 3. The dependency degree of B to D is defined as the ratio of consistent objects:

$$\gamma_B(D) = \frac{CARD(POS_B(D))}{CARD(U)}, \tag{6}$$

where $CARD(U)$ is number of instances of the universe (sample set), and $CARD(POS_B(D))$ is the number of positive region objects.

The dependency degree reflects the describing capability of all attributes in B. Meanwhile, it can also be considered as the significance of attributes in B to approximate decision D.

3. Online dependency feature selection framework

In this section, we first give a definition on online streaming feature selection for class imbalance data. Then we introduce three evaluation criteria of "maximal-dependency, maximal-relevance and maximal-significance" based on the dependency between condition features and decision classes. Our new online feature selection framework and a new algorithm based on it by using the nearest K-neighbors information will be presented sequentially.

3.1. Problem statement

An online streaming feature selection framework can be defined as $F = (U, C \cup D, f, t)$, where U is a non-empty finite set of objects, C is the condition attribute set, and D is the decision attribute set. Let $C = [x_1, x_2, \ldots, x_n]^T \in R^{n\times d}$ consist of n samples over a d-dimensional feature space $F = [f_1, f_2, \ldots, f_d]^T \in R^d$. Let $D = [y_1, y_2, \ldots, y_n]^T \in R^{n\times 1}$ consist of n samples over the class label (decision feature space) L. Let $L = \left[l_{small}, l_{large}\right]^T \in R$ denote the class label vector, where there are far more instances in l_{large} than in l_{small}. Given U, C and D, at each time stamp t, we get a feature f_t of $C \cup D$ without knowing the exact number of d in advance. The problem is to derive a mapping f: $C' \to L$ at each time stamp t, which is as good as possible using a subset of features that have arrived so far.

Regarding the high-dimensional class imbalance data, traditional online feature selection approaches can not look after the imbalance data distribution and tend to be overwhelmed by the large classes and ignore the small ones. When the skewness of the class distribution in a data set is drastically high, a naive online feature selection approach can get very high accuracy on the whole data set by classifying all instances to the large class. Nevertheless, this does not make much sense because all instances in the small class are classified falsely. Meanwhile, in real-world applications, such as in the field of medical diagnosis, people pay more attention to those rare abnormal cases than the common normal cases. Thus, it is nontrivial for us to provide a new solution to the problem of class imbalance for online feature selection. In the next section, we will introduce three evaluation criteria for our new online feature selection framework.

3.2. Evaluation criteria of maximal-dependency, maximal-relevance and maximal-significance

In order to remove irrelevant and redundant features in the process of feature selection, we introduce three evaluation criteria as follows [53].

3.2.1. Maximal-dependency

Let $C = C_1, C_2, \ldots, C_m$ denote the set of m condition features of a given data set. In terms of neighborhood rough sets, the task of feature selection is to find a feature subset $S \subseteq C$ with $d < m$ features which have the largest dependency $\mathbb{D}$ on the decision attribute set D. It can be represented as Eq. 7.

$$\mathbf{Max}\mathbb{D}(S, D), \mathbb{D} = \gamma_{\{C_i, i=1,2,\ldots d\}}(D), \tag{7}$$

where $\mathbb{D} = \gamma_{\{C_i, i=1,2,\ldots d\}}(D)$ indicates the dependency between the feature subset S and the target class label D as shown in Eq. 6.

Theoretically, the maximal-dependency is the best evaluation criterion in feature selection with neighborhood rough sets. However, it is hard to generate the resultant equivalence classes using the maximal-dependency in high-dimensional spaces due to the following two challenges: the number of samples is often insufficient and the generation of resultant equivalence classes is usually an ill-posed problem [5]. In addition, we can not only use maximal-dependency for online streaming feature selection because we just get one feature at each time stamp and we do not know the whole feature space in advance.

3.2.2. Maximal-relevance

Maximal-relevance is to search features which approximate $\mathbb{D}(S, D)$ using Eq. (7) with the mean value of all dependency values between each individual feature C_i and the target class label D:

$$\mathbf{Max}\mathbb{R}(S, D), \mathbb{R} = \frac{1}{|S|}\sum_{C_i \in S} \gamma_{C_i}(D). \tag{8}$$

The dependency among selected features according to maximal-relevance could have rich redundancy. For example, if two features highly depend on each other, both of them are in the candidate feature subset, and we remove one of them, the respective class discriminative power would not change a lot. Thus, the evaluation criterion of maximal-relevance can select features with a high dependency to the decision classes, but it can not remove redundancy in the selected feature subset.

3.2.3. Maximal-significance

Definition 4. Given condition attribute set C and a decision attribute set D, a feature $A \in C$, the significance of A is defined as:

$$\sigma_C(A, D) = \gamma_C(D) - \gamma_{C-A}(D). \tag{9}$$

With the significance of each feature to a feature set, we can measure each feature's importance in the selected candidate subset. The maximal-significance condition can select mutually exclusive features as follows:

$$\mathbf{Max}\mathbb{S}(S, D), \mathbb{S} = \frac{1}{|S|}\sum_{C_i \in S}\{\sigma_S(C_i, D)\}. \tag{10}$$

For online feature selection, the features flow in one by one over time. We can not test all combinations of candidate features to maximize the dependency of the selected feature set as Eq. (7). However, we can use the "maximal-relevance" condition to select relevant features and discard irrelevant features at first. Then we use the "maximal-significance" criterion to remove nonsignificant features in selected feature set. The "Maximal-dependency" criterion will be used as the final goal of selecting the feature set with maximal dependency. In the next section, based on the aforementioned three measures, we will propose a new online feature selection framework.

3.3. Our framework

Feature selection aims to derive a mapping function from samples to classes that is "as good as possible". This can be treated as a decision system problem [52].

Definition 5. Let $DS =< U, A, D >$ denote a decision system, where U is the sample set, called the universe, A is the condition attribute set and D is the decision attribute set.

Let $\gamma_{S\subset A}(D)$ denote the dependency degree of S to D, where S is a subset of A. If there is only one feature in S, we can define the dependency degree of a single feature f as $\gamma_f(D)$. Thus,

with the "maximal-dependency" evaluation criterion, the problem of feature selection can be treated as finding a subset $S \subset A$ where $\gamma_S(D)$ achieves the maximum, as shown in Eq. (7).

For high dimensional data sets, there are always many irrelevant features. At time stamp j, if f_j is irrelevant, we can just discard this feature. An irrelevant feature means a low dependency to the decision class.

Theorem 1. *Suppose at time stamp t_{j-1}, the selected feature set is S_{j-1}. Let $\mathbb{R}_{j-1} = \frac{1}{|S_{j-1}|}\sum_{f_i \in S_{j-1}} \gamma_{f_i}(D)$ and $\forall \gamma_{f_i \in S_{j-1}} > \alpha$. At time stamp t_j, the new arriving feature is f_j. If $\gamma_{f_j}(D) < \alpha$ and we add f_j into S_{j-1}, then $\mathbb{R}_j < \mathbb{R}_{j-1}$.*

Proof 1. Let $|S_{j-1}| = n_{j-1}$ and $\mathbb{R}_{j-1} = r_{j-1}$. It is obvious that $\sum_{f_i \in S_{j-1}} \gamma_{f_i}(D) = r_{j-1} \times n_{j-1}$. $\because \forall \gamma_{f_i \in S_{j-1}} > \alpha$, $\therefore r_{j-1} > \alpha$. If we add f_j into S, then $S_j = S_{j-1} \cup f_j$.
$\mathbb{R}_j = \frac{1}{|S_j|}\sum_{f_i \in S_j} \gamma_{f_i}(D) = \frac{1}{n_{j-1}+1}(n_{j-1} \times r_{j-1} + \gamma_{f_j}(D))$
$= \frac{n_{j-1}}{n_{j-1}+1} r_{j-1} + \frac{1}{n_{j-1}+1}\gamma_{f_j}(D) = r_{j-1} + \frac{1}{n_{j-1}+1}(\gamma_{f_j}(D) - r_{j-1})$.
$\because \gamma_{f_j}(D) < \alpha < r_{j-1}$, $\therefore \gamma_{f_j}(D) - r_{j-1} < 0$, $\therefore \mathbb{R}_j < \mathbb{R}_{j-1}$.

Thus, with the "maximal-dependency" evaluation criterion, if we add a feature with a low dependency ($\gamma_{f_j}(D) < \alpha$) into the selected feature set S (while all other features in S have the dependency bigger than α), the mean value of all dependency values between each individual feature f_i ($f_i \in S$) and the target class label D will decrease. For the arriving feature f_j, if $\gamma_{f_j}(D)$ is smaller than the threshold α, it will be discarded for efficiency.

Theorem 2. [54] *Suppose B is a subset of conditional features, f is an arbitrary conditional attribute that belongs to the dataset, and D is the set of decision attributes. Then $\gamma(B \cup f, D) \geq \gamma(B, D)$.*

Proof 2. The proof of this theorem is available in [54].

Theorem 3. *Suppose at time stamp t_{j-1}, the selected feature set is S_{j-1}. At time stamp t_j, the new arriving feature is f_j. If $\gamma_{f_j} > \gamma_{S_{j-1}}$, then $\gamma_{f_j} > \gamma_{\forall f_i \in S_{j-1}}$.*

Proof 3. Let $S_{j-1} = \{f', f'', \ldots, f^i\}$, and let $\gamma(S_{j-1}, D)$ denote as $\gamma_{S_{j-1}}$ for short. From Theorem 2, we can deduce that $\gamma_{S_{j-1}} \geq \gamma_{\{S_{j-1}-f^i\}} \geq \gamma_{\{S_{j-1}-f^i-f^{i-1}\}} \geq \cdots \geq \gamma_{f'}$. $\therefore \gamma_{S_{j-1}} \geq \gamma_{\forall f_i \in S_{j-1}}$.
$\because \gamma_{f_j} > \gamma_{S_{j-1}}$, $\therefore \gamma_{f_j} > \gamma_{\forall f_i \in S_{j-1}}$.

If the new arriving feature f_j has a bigger dependency than the selected feature set S_{j-1}, γ_{f_j} will be bigger than all the features in S_{j-1}. It is obvious that if we add f_j into S_{j-1}, $\mathbb{R}_j$ will decrease. If we replace the selected feature set S_{j-1} with f_j ($S_j = \{f_j\}$), then, $\mathbb{R}_j = r_{f_j}$. According to Theorem 3, $\gamma_{f_j} > \gamma_{\forall f_i \in S_{j-1}}$, then $\gamma_{f_j} > \mathbb{R}_{j-1}$. Thus, $\mathbb{R}_j > \mathbb{R}_{j-1}$. With the "maximal-dependency" evaluation criterion, we replace the current selected feature set with a new arriving feature when the feature has a higher dependency degree. That is, with the new arriving feature f_j at the *jth* time stamp, if $\gamma_{f_j}(D)$ is bigger than $\gamma_{S_{j-1}}(D)$, we will just retain f_j as S_j and discard all of the selected features in S_{j-1}.

If the arriving feature f_j at time stamp j satisfies the "maximal-relevance" constraint ($\gamma_{f_j}(D) > \alpha$), the dependency of f_j is equal to or lower than the dependency of the currently selected feature set S_{j-1} ($\gamma_{f_j}(D) \leq \gamma_{S_{j-1}}$). In order to decide whether adding feature f_j into S, we need to compute the significance of f_j. Thus, we give the definition of significance measures in online feature selection as follows.

Definition 6. Given the arriving feature f_j and selected feature subset S, the significance of f_j can be denoted as:

$$Sig(f_j) = \gamma_{S \cup f_j}(D) - \gamma_S(D). \tag{11}$$

With the "maximal-significance" evaluation criterion, for the arriving feature f_j, if and only if $Sig(f_j) > 0$, f_j will be added onto the candidate feature set. This means, we only select those features that enable increasing the dependency degree of the selected subset to the decision attribute.

Based on the above analysis, we propose a new Online Feature Selection framework based on the dependency between condition features and the decision classes, named OFSD as shown Algorithm 1. With the "maximal-dependency, maximal-relevance and maximal-significance" evaluation criteria, OFSD can select features with high correlation, high dependency and low redundancy.

Algorithm 1 Our OFSD framework.

Require:
S_{t_i}: the selected feature set at time stamp t_i;
α: threshold of the dependency between a single feature and the decision attribute;
Ensure:
S: the selected feature set
1: Initialize S to {};
2: **Repeat**
3: Get a new feature f_i at time t_i;
4: Calculate the dependency of f_i with classes: γ_{f_i}
5: IF $\gamma_{f_i} < \alpha$
6: Discard f_i ; and go to Step 17;
7: End IF
8: IF $\gamma_{f_i} > \gamma_{S_{t_{i-1}}}$
9: $S_{t_i} = f_i$
10: ELSE
11: IF $\gamma_{S_{t_{i-1}} \cup f_i} - \gamma_{S_{t_{i-1}}} > 0$
12: $S_{t_i} = S_{t_{i-1}} \cup f_i$
13: ELSE
14: $S_{t_i} = S_{t_{i-1}}$
15: END IF
16: END IF
17: **Until** no more features are available;
18: **return** S;

At time stamp i, we calculate the dependency of arriving feature f_i at Step 5 and compare it with the threshold α. It can greatly reduce the running time by discarding features with a lower dependency. If γ_{f_i} is bigger than α, we will compare it with the dependency of the selected feature set $\gamma_{S_{t_{i-1}}}$ at Step 8. If it is also larger, all the features in S will be removed and only the feature f_i is maintained. Otherwise, we will calculate the significance of f_i at Step 11. Only if the significance of f_i is bigger than 0 (in other words, the arriving feature f_i can increase the dependency degree of the selected subset), f_i will be added onto the candidate subset. Otherwise, f_i will be discarded.

The key technique in our framework is to calculate the dependency between condition features and decision attributes. In the following section, we will give a framework for dependency computation.

3.4. Dependency computation

In this paper, we develop a refined method based on the neighborhood rough set theory to calculate the dependency between features and classes for class imbalance data. The proposed dependency computation method is shown as Algorithm 2.

At Steps 3–6, we calculate the CARD value of each instance x_i and get the sum for the final dependency degree. The CARD value ranges from 0 to 1, denoted as the consistency of x_i's class attribute with its neighbors' class attributes. Theoretically, we can use any

Algorithm 2 Dependency computation.

Require:
X_S: sample values on feature set S;
R: neighborhood relation,
Ensure:
dep_S: dependency on feature set S
1: $card_S$: the number of positive samples S, initialized to 0
2: $card_U$: number of instances of X_S (number of universe, N)
3: FOR each x_i in X_S
4: find the neighbor samples of x_i on R as $S_R(x_i)$
5: calculate the card value of x_i as $CARD(S_R(x_i))$
6: $card_S = card_S + CARD(S_R(x_i))$
7: END FOR
8: $dep_S = card_S/card_U$
9: **return** dep_S;

neighborhood relation to calculate the dependency of feature set S. The value of $Card(S_R(x_i))$ indicates the distribution of classes near by x_i.

We can also use different methods for calculating the value of CARD, such as:

- **Card_Consistency**: If all the classes of samples in $S_R(x_i)$ are the same as the class of x_i, then $Card(S_R(x_i)) = 1$; else $Card(S_R(x_i)) = 0$.
- **Card_Weight**: Assume Num_P is the number of samples in $S_R(x_i)$ which have the same class as x_i, and Num_S is the size of set $S_R(x_i)$.
  If $\frac{Num_P}{Num_S} > \gamma$, then $Card(S_R(x_i)) = \frac{Num_P}{Num_S}$; else $Card(S_R(x_i)) = 0$, where γ is a threshold and the default value is set to 0.5.
- **Card_Distance**: Assume Num_P is the number of samples in $S_R(x_i)$ which have the same class as x_i, and $Sum(dis_p)$ is the sum of distances to x_i from these Num_P samples. Num_N is the number of samples in $S_R(x_i)$ which have the different class to x_i and $Sum(dis_n)$ is the sum of distances to x_i from these Num_N samples. Num_S is the size of set $S_R(x_i)$.
  If $\frac{Num_P}{Num_S} \geq \gamma_1$, then $Card(S_R(x_i)) = 1$;
  If $\frac{Num_P}{Num_S} \leq \gamma_2$, then $Card(S_R(x_i)) = 0$;
  If $\gamma_2 < \frac{Num_P}{Num_S} < \gamma_1$, and if $\frac{Sum(dis_p)}{Num_N} > \frac{Sum(dis_n)}{Num_N}$, then $Card(S_R(x_i)) = 1$, else $Card(S_R(x_i)) = 0$, where γ_1 and γ_2 are thresholds that control the ratio of Num_P to Num_N respectively.

In a class imbalanced data set, the number of instances in the large class is far more than that in the small class. In order to find the features which can get higher separability between the large class and the small class, we should treat instances from different classes with different strategies. Thus, we propose the CARD function **Card_Imbalanced** for dependency computation in Algorithm 3.

At Steps 3–11, we can see the difference between C_{large} and C_{small} instances. If x_i belongs to the large class, only if all the neighbors belong to C_{large}, we will set the CARD value to 1. Otherwise, if x_i is in the small class, we calculate the ratio of the number of instances with C_{small} to the total number of neighbors as the CARD value. In order to prevent instances in the small class from being overwhelmed by the instances in the large class, we strengthen consistency constraints of the large class and weaken consistency constraints of the small class.

Algorithm 3 Card_Imbalanced function.

Require:
C_{x_i}: the class label of x_i;
R: neighborhood relation.
Ensure:
$Card_{x_i}$: the CARD value of x_i
1: N_R : the number of neighbors on R
2: N_p : the number of neighbors with the same class label of x_i on R
3: IF C_{x_i} == C_{large}
4: IF N_R == N_p
5: $Card_{x_i} = 1$
6: ELSE
7: $Card_{x_i} = 0$
8: End IF
9: ELSE //C_{x_i} == C_{small}
10: $Card_{x_i} = N_p/N_R$
11: END IF
12: **return** $Card_{x_i}$;

3.5. The K-OFSD algorithm

3.5.1. Dependency function of K-OFSD

We use $k_S(x_i)$ to calculate the dependency and propose the Online Feature Selection algorithm based on dependency in K nearest neighbors, named K-OFSD. Based on the OFSD framework, the difference lies in the dependency calculation denoted as Dep_K. More details of Dep_K are as shown in Algorithm 4.

Algorithm 4 Dep_K of K-OFSD.

Require:
K: the number of nearest samples ;
X_S: sample values on feature set S;
Ensure:
dep_S: dependency of S
1: $Card_S$: the sum of every sample's CARD value on S, initialized to 0
2: $Card_U$: instance number of X_S (number of universe, N)
3: FOR each x_i in X_S
4: C_{x_i}: the class label of x_i
5: Find the K nearest samples of x_i as $S_K(x_i)$
6: $Card_S = Card_S +$ **Card_Imbalanced**$(C_{x_i}, S_K(x_i))$
7: END FOR
8: $dep_S = Card_S/Card_U$
9: **return** dep_S;

In Dep_K, Step 5 finds K nearest neighbors of each x_i as the neighborhood relation instance set $S_K(x_i)$. Step 6 uses **Card_Imbalanced** function for card value calculation. The time complexity of Dep_K is $O(|X_S|^2)$. The complete algorithm of K-OFSD is shown in Algorithm 5.

The default value of α is set as 0.5 which means more than half of the K nearest neighbors for each object x_i have the same class label as x_i. The parameter K that controls the number of neighbors affects the dependency value of feature f_j at time stamp j. An important task in K-OFSD is how to choose a good and unified value K for different data sets. Actually, we will see that K-OFSD is not very sensitive to parameter K. More details will be provided in the experimental analysis in Section 4.2.

3.5.2. The time complexity of K-OFSD

The time complexity of K-OFSD depends on the dependency function Dep_K. Suppose the data set is DS, the number of instances

Algorithm 5 K-OFSD.

Require:
S_{t_i}: the selected feature set at time stamp t_i;
K: the number of nearest samples;
α: threshold of the dependency between a single feature and the decision attribute (default value is 0.5);
Ensure:
S: the selected feature set
Initialize S to {};
Repeat
  Get a new feature f_i at time t_i;
  Calculate the dependency of f_i as: $Dep_K(X_{f_i})$
  IF $Dep_K(X_{f_i}) < \alpha$
    Discard f_i ; and go to Step 17;
  End IF
  IF $Dep_K(X_{f_i}) > Dep_K(S_{t_{i-1}})$
    $S_{t_i} = f_i$
  ELSE
    IF $Dep_K(S_{t_{i-1}} \cup f_i) > Dep_K(S_{t_{i-1}})$
      $S_{t_i} = S_{t_{i-1}} \cup f_i$
    ELSE
      $S_{t_i} = S_{t_{i-1}}$
    END IF
  END IF
Until no features are available;
return S;

Table 2
Experimental data sets.

Data set	Instances(Train/Test)	Features	Class(min./maj.)	Ratio
DLBCL	77(39/38)	6285	19 / 58	3.05
LUNG	181(91/90)	12533	31 / 150	4.84
LYM	62(31/31)	4026	9 / 53	5.89
GLIOMA	50(25/25)	4433	7 / 43	6.14
SRBCT	83(42/41)	2308	11 / 72	6.55
LUNG2	203(102/101)	3312	17 / 186	10.94
CAR	174(87/87)	9182	7 / 167	23.85

in DS is N, the number of features in DS is F, and the number of neighbors used in Dep_K is K. According to Section 3.5.1, the time complexity of Dep_K is $O(|N|^2)$. At time stamp t, a new feature f_t is present to the K-OFSD algorithm. Assume S_t is the selected feature subset at this time. K-OFSD first calculates the dependency of f_t, and the time complexity is $O(|N|^2)$. If the dependency of f_t is smaller than $\alpha(0.5)$, f_t will be discarded. Otherwise, we will calculate the dependency of S_t and compare it with $Dep_K(f_t)$. This time complexity is also $O(|N|^2)$. If the dependency of f_t is bigger than $Dep_K(S_t)$, we replace S_t with f_t and go on to the next feature. Otherwise, we will calculate the dependency of $S_t \cup f_t$ and compare it with $Dep_K(f_t)$. Then we can decide on whether adding f_t onto the selected feature subset or discarding it. The time complexity is $O(|N|^2)$.

In sum, the whole time complexity of K-OFSD is $O(|F||N|^2)$. We can use more effective dependency calculation methods to improve the time performance in our future work.

4. Experimental results

4.1. Experiment setup

4.1.1. Data sets

In this section, we apply the proposed online feature selection algorithm on seven high dimensional DNA microarray data sets [55,56] as shown in Table 2.

In our experiments, we divide some data sets into one-versus-rest problems and adapt these data sets to imbalanced binary classification. Detailed descriptions of these data sets are as follows.

- DLBCL: This diffuse large B-cell lymphoma data set has a total of 77 samples in two classes, diffuse large B-cell lymphomas (DLBCL) and follicular lymphoma (FL), which have 58 and 19 instances, respectively. Each instance contains 6285 features.
- LUNG: The lung cancer data set contains a total of 181 samples in two classes, which have 31 malignant and 150 normal samples, respectively.
- LYMPHOMA: The lymphoma data set contains a total of 62 samples in three classes which have 11, 9 and 42 samples, respectively. To adapt this data set to imbalanced binary classification, we studied a class of 9 instances versus the rest.
- GLIOMA: The GLIOMA data set contains a total of 50 samples in four classes, cancer glioblastomas (CG), non-cancer glioblastomas (NG), cancer oligodendrogliomas (CO) and non-cancer oligodendrogliomas (NO), which have 14, 14, 7, 15 samples, respectively. To adapt this data set to imbalanced binary classification, we studied class CO (7 instances) versus the rest.
- SRBCT: The SRBCT data set contains a total of 83 samples in four classes, the Ewing family of tumors (EWS), Burkitt lymphoma (BL), neuroblastoma (NB) and rhabdomyosarcoma (RMS). Every sample in this data set contains 2308 gene expression values. Among the 83 samples, 29, 11, 18, and 25 samples belong to classes EWS, BL, NB and RMS, respectively. To adapt this data set to imbalanced binary classification, we studied class BL (11 instances) versus the rest.
- LUNG2: The LUNG2 data set contains a total of 203 samples in five classes, adenocarcinomas, squamous cell lung carcinomas, pulmonary carcinoids, small-cell lung carcinomas and normal lung, which have 139, 21, 20, 6, 17 samples, respectively. To adapt this data set to imbalanced binary classification, we studied class normal lung (17 instances) versus the rest.
- CAR: The CAR data set contains a total of 174 samples in eleven classes, prostate, bladder/ureter, breast, colorectal, gastroesophagus, kidney, liver, ovary, pancreas, lung adenocarcinomas, and lung squamous cell carcinoma, which have 26, 8, 26, 23, 12, 11, 7, 27, 6, 14, 14 samples, respectively. To adapt this data set to imbalanced binary classification, we studied class liver (7 instances) versus the rest.

In our experiments, we use two basic classifiers, KNN and SVM in Matalab R2015b to evaluate a selected feature subset. We randomly select 1/2 of the samples for training and the rest for testing. The training and testing data sets have the same ratio of class imbalance. All experimental results are averaged over 20 runs and are conducted on a PC with Intel(R) i5-3470S, 2.9 GHz CPU, and 8 GB memory.

4.1.2. Performance metric for class imbalance data

The most frequently used metric for classification is predictive accuracy. However, it is not appropriate for classification of imbalanced data sets [26]. For instance, if the data set has 1% of the small-class instances and 99% of the large-class instances, and if a simple method classifies all samples to the large class, the accuracy also can be up to 99%. In the classification for class imbalance data, we should take the small class with a higher misclassification cost [26].

In this work, we use the G-MEAN as the main performance metric [57]. G-MEAN= $\sqrt{\frac{TP}{TP+FN} * \frac{TN}{TN+FP}}$, where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives and FN is the number of false negatives.

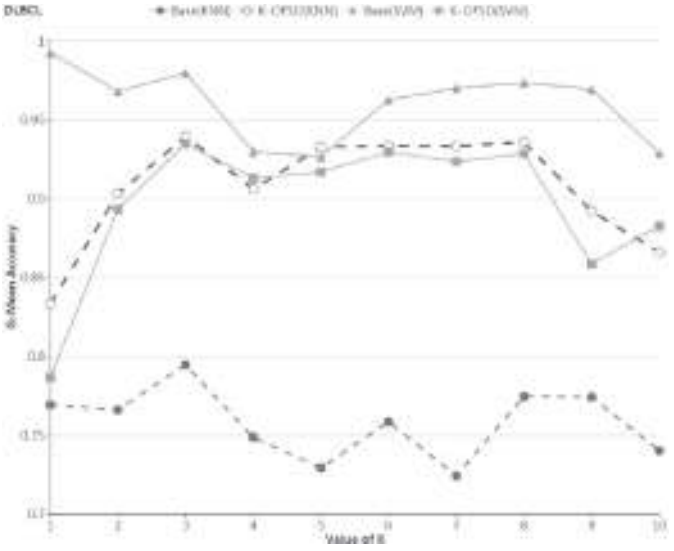

Fig. 2. Accuracy on DLBCL varying with K.

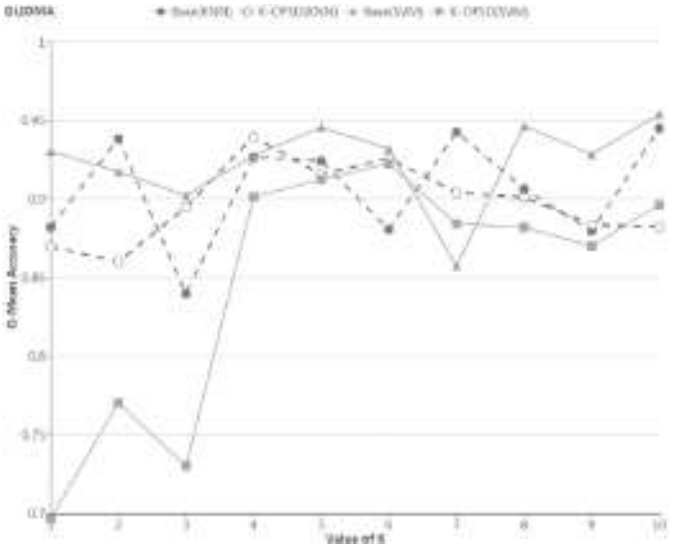

Fig. 3. Accuracy on GLIOMA varying with K.

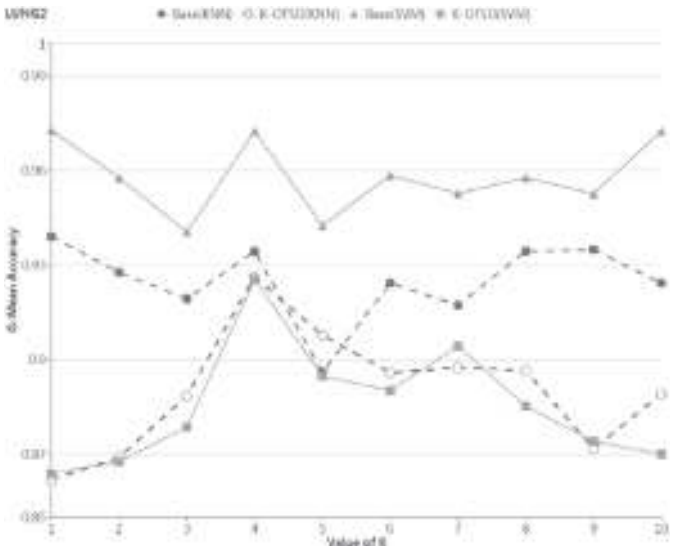

Fig. 4. Accuracy on LUNG2 varying with K.

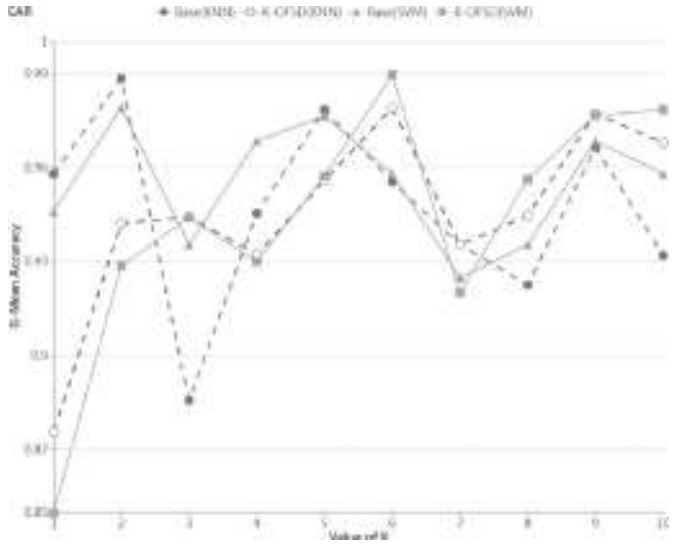

Fig. 5. Accuracy on CAR varying with K.

4.2. Parameter analysis for K-OFSD

In this section, we will discuss the effect of parameter K in K-OFSD which controls the number of neighbors used in function Dep_K.

Figs. 2, 3, 4, 5 show the experimental results of our algorithm on four data sets DLBCL, GLIOMA, LUNG2 and CAR varying with values of K from 1 to 10. Fig. 6 and Fig. 7 show the number of selected features and running time on these four data sets varying with different values of K. In these experiments, we select SVM and KNN as the basic classifiers, and the value of k in KNN is set to 1.

From Fig. 2 to Fig. 7, we have the following observations:

- From Fig. 2 to Fig. 5, the accuracy of K-OFSD with KNN is similar to with SVM. Meanwhile, the baselines KNN and SVM present a large gap in the accuracy, especially on data sets of DLBCL and LUNG2. This indicates that the features selected by K-OFSD can fit both KNN and SVM well.
  Besides, on DLBCL, our algorithm can perform better with any K value compared to the baseline using KNN, and it is inferior

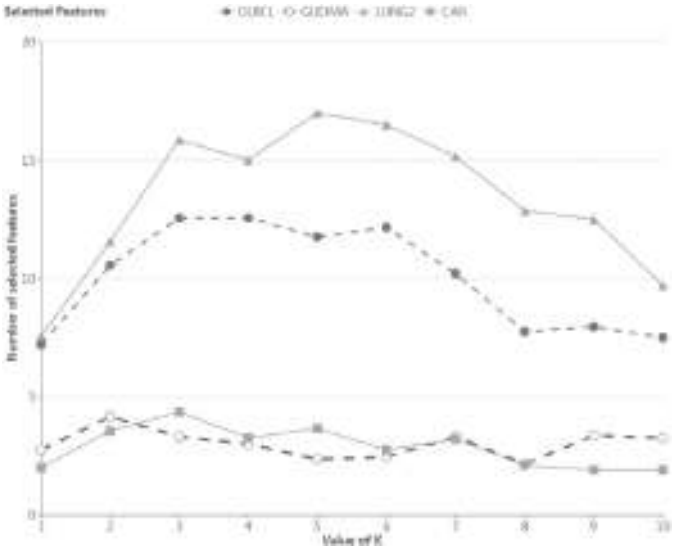

Fig. 6. Number of selected features varying with K.

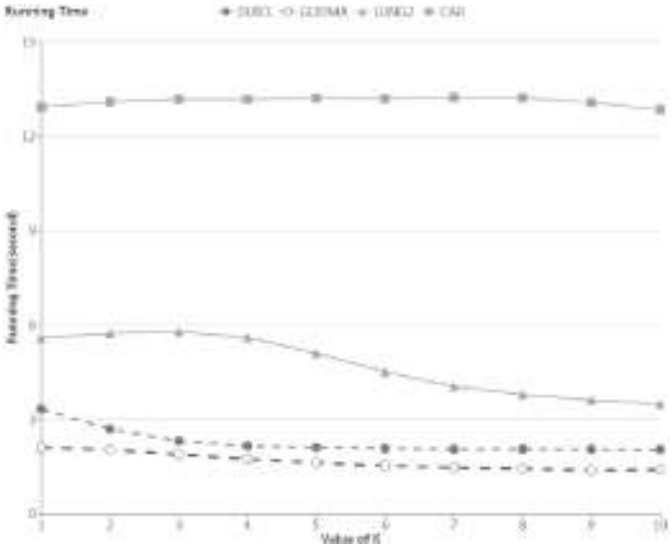

Fig. 7. Running time varying with K.

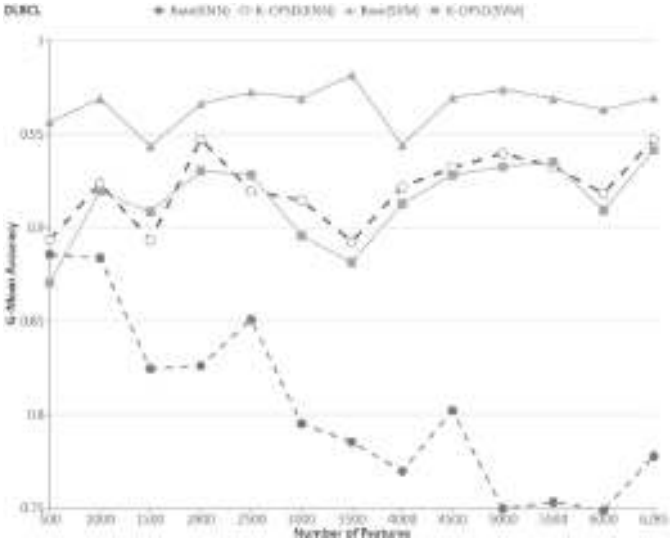

Fig. 8. Accuracy on DLBCL varying with numbers of features.

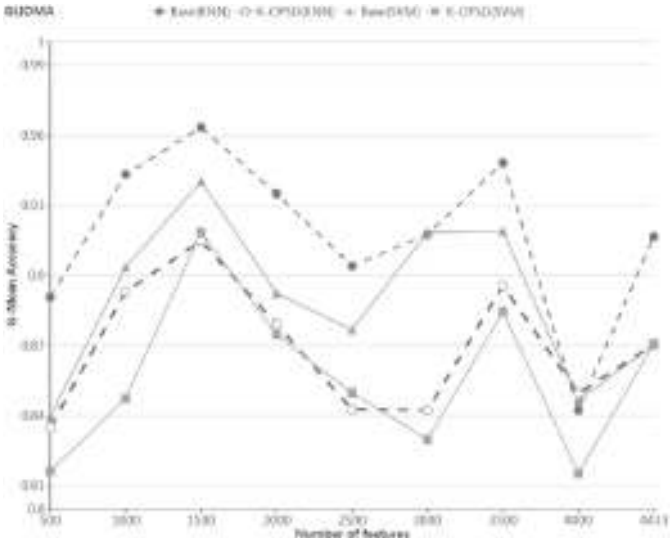

Fig. 9. Accuracy on GLIOMA varying with numbers of features.

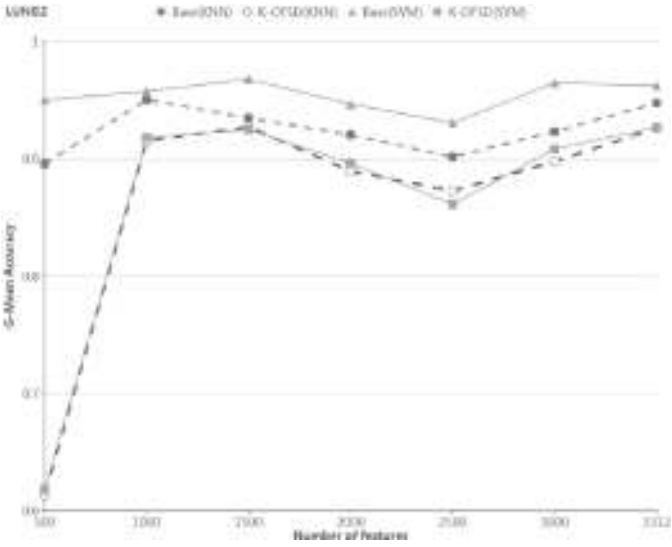

Fig. 10. Accuracy on LUNG2 varying with numbers of features.

to the baseline using SVM. On LUNG2, the accuracy of our algorithm with any K value is lower than the baselines using both KNN and SVM. On GLIOMA and CAR, our algorithm provides a higher accuracy or a lower accuracy with different K values compared to the baselines. This indicates the number of neighbors has little impact on data sets DLBCL and LUNG2, but has significant impact on data sets GLIOMA and CAR. This is closely related to the class distribution of each data set.

- From Fig. 6, the number of selected features on data sets CAR and GLIOMA is always around 2 and 3 despire varying with different values of K. On data sets DLBCL and LUNG2, the number of selected features increases first and then decreases. The number reaches the maximum when K is around 5 and 6. This indicates, for some data sets, the K value does not affect much the final selected number of features. For others, a suitable K is necessary for good performance. The main reason is the sparsity of samples in different data sets.
- From Fig. 7, the running time of our algorithm is irrelevant to the value of K.
- In sum, the accuracy fluctuates relatively large on some data sets (eg. GLIOMA and CAR) in comparison with others (eg. DLBCL and LUNG2) when varying with values of K. When $K > 5$, the fluctuation becomes weak on predictive accuracy for all of the four data sets. This indicates that a relatively large value of K ($K > 5$) is suitable for stable results.

From the above experimental analysis, we can see that K-OFSD is sensitive to the parameter K for some data sets, and a relatively large value of K can provide a stable predictive accuracy for all data sets. We select $K = 7$ below for the global optimum in our experiments.

4.3. The influence of numbers of features

In this section, we test the influence of numbers of features on our method. Figs. 8, 9, 10, 11 show the experimental results of predictive accuracy of our algorithm on four data sets DLBCL, GLIOMA, LUNG2 and CAR varying with numbers of features. Fig. 12 and Fig. 13 show the number of selected features and running time on these four data sets varying with numbers of features. In these experiments, we set $K = 7$ for our algorithm and select SVM and KNN as the basic classifiers. The value of k in KNN is set to 1.

From Fig. 8 to Fig. 13, we have the following observations:

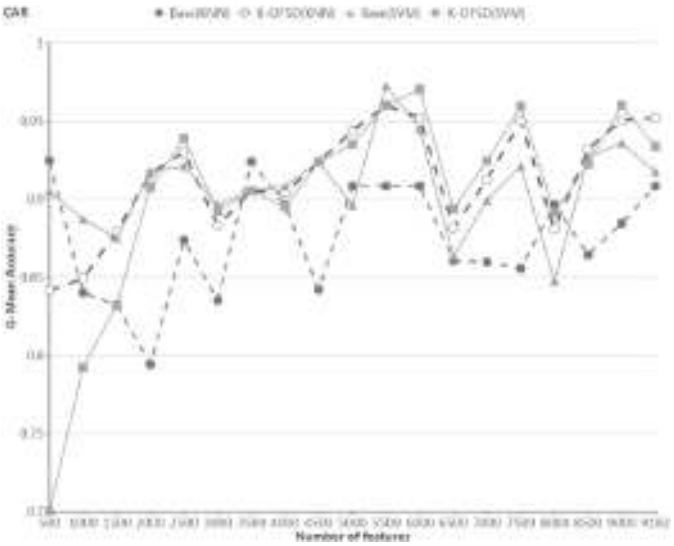

Fig. 11. Accuracy on CAR varying with numbers of features.

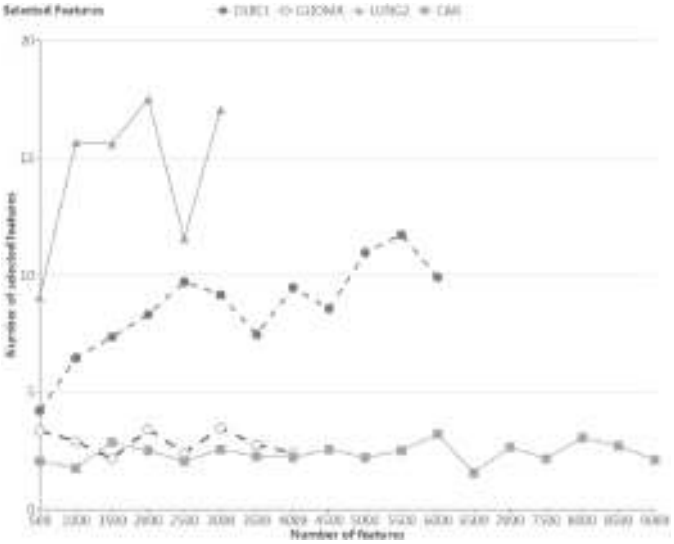

Fig. 12. Number of selected features varying with numbers of features.

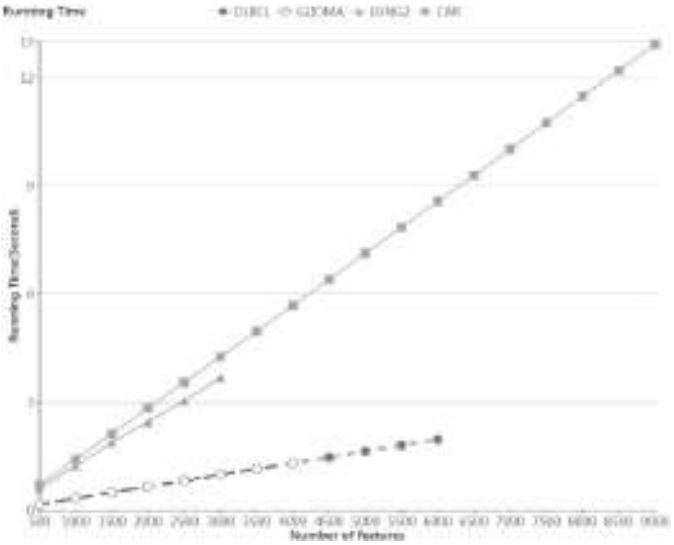

Fig. 13. Running time varying with numbers of features.

Table 3
G-MEAN predictive accuracy using the KNN classifier.

Data set	K-OFSD	Fisher	S2N	PCC	ReliefF	MI
DLBCL	**0.9381**	0.8111	0.9302	0	0.8451	0.9171
LUNG	0.9925	0.983	**0.9959**	0.983	0.9829	0.9898
LYM	0.9844	0.9788	**0.9922**	0.9384	0.9224	0.981
GLIOMA	**0.8817**	0.5751	0.8506	0	0.7275	0.8334
SRBCT	**0.9971**	0.9583	0.9789	0.9894	0.9525	0.9565
LUNG2	**0.9908**	0.9586	0.9715	0.9844	0.9566	0.978
CAR	**0.9449**	0.7256	0.9234	0	0.9065	0.9443
AVERAGE	**0.961**	0.856	0.949	0.556	0.899	0.943

- From Fig. 8 to Fig. 11, the accuracy of K-OFSD with KNN performs similarly to SVM with different numbers of features. On the contrary, the baselines KNN and SVM present a gap in the predictive accuracy. Especially on data set DLBCL, with the increasing of the number of features, the predictive accuracy of the baseline with KNN is decreasing. The main reason is there are more and more irrelevant and redundant features flowing in. This indicates that the features selected by K-OFSD have high separability and can fit both KNN and SVM well.
  In addition, on data sets GLIOMA, LUNG2 and CAR, when the number of features is lower than 1500, the performance of K-OFSD is inferior to the baselines. As more and more features arrive, the predictive accuracy of K-OFSD becomes competitive with and then superior to the predictive accuracy of baselines. This also demonstrates K-OFSD can select highly discriminative features from massive irrelevant and redundant streaming features.
- From Fig. 12, on data sets GLIOMA and CAR, the mean number of selected features is around 2 and 3 varying with different numbers of features. On data sets DLBCL and LUNG2, the number of selected features increases with the number of features. This indicates that the number of selected features depends on specific data sets and it is different from each other.
- From Fig. 13, there is a linear relationship between the running time of our algorithm and the number of features. Besides, the linear slope is associated with the number of training instances.

From the above experimental analysis, we can conclude that K-OFSD can select features with high separability varying with different numbers of features. The number of selected features is related to specific data sets and there is a linear relationship between the running time and the number of features.

4.4. K-OFSD vs. traditional imbalanced feature selection methods

In this section, we compare K-OFSD with five traditional imbalance feature selection methods, including: ReliefF [2], Fisher Score [3], S2N (signal-to-noise correlation coefficient) [30], PCC (Pearson Correlation Coefficient) [30] and MI (mutual information) [4].

All these algorithms are implemented in MATLAB. The K value of ReliefF is set to 7, the same as K-OFSD. None of these five traditional feature selection methods can handle the scenario of feature streaming in an online manner. Thus, we rank all features from high to low and select the same number of features for K-OFSD. We evaluate K-OFSD and all competing ones on the G-MEAN predictive accuracy.

Tables 3 and 4 summarize the G-MEAN predictive accuracy of K-OFSD against the other five competing algorithms using the basic classifiers of KNN (k=1) and SVM. Table 5 shows the mean number of selected features on different data sets.

From Tables 3, 4 and 5, we have the following observations.

- K-OFSD vs. Fisher. K-OFSD outperforms Fisher on all these data sets with both KNN and SVM. On data sets GLIOMA and CAR,

Table 4
G-MEAN predictive accuracy using SVM as the base calssifier.

Data set	K-OFSD	Fisher	S2N	PCC	ReliefF	MI
DLBCL	**0.9325**	0.812	0.9194	0	0.8388	0.9136
LUNG	**0.9919**	0.9695	0.9892	0.9898	0.9817	0.9898
LYM	0.9784	0.9788	**0.9922**	0.9559	0.9442	0.9517
GLIOMA	**0.8498**	0.5037	0.8434	0	0.6952	0.7482
SRBCT	**0.9942**	0.9584	0.9894	0.9894	0.9563	0.9459
LUNG2	**0.9769**	0.957	0.9635	**0.9769**	0.963	0.971
CAR	**0.9449**	0.7256	0.9234	0	0.9064	0.9261
AVERAGE	**0.953**	0.844	0.946	0.559	0.898	0.921

Table 5
The mean number of selected features.

Data set	K-OFSD/Fisher/S2N/PCC/ReliefF/MI
DLBCL	10.9
LUNG	25.7
LYM	6.7
GLIOMA	3.1
SRBCT	5.2
LUNG2	21.3
CAR	1.7

Fisher only gets the predictive accuracy of 0.5 and 0.7 respectively using KNN and SVM. Fisher computes the importance of each feature by calculating the difference of that feature's mean values for the two classes. It measures features individually and it can not consider all selected feature set as a whole. The results demonstrate that K-OFSD is superior to Fisher on imbalance data.

- K-OFSD vs. S2N. K-OFSD performs better than S2N on five of the seven data sets. S2N performs stably on all of the seven data sets and gets the highest accuracy on data set LYM and LUNG. This result verifies the conclusion in [30], that "S2N is a great candidate algorithm for feature selection in most applications on imbalance data". Overall, S2N is very competitive to K-OFSD on predictive accuracy.
- K-OFSD vs. PCC. K-OFSD outperforms PCC on six of the seven data sets. PCC gets the predictive accuracy 0 on data sets DLDBC, GLIOMA and CAR with both KNN and SVM. This is because PCC can not classify the instances of the small class correctly on these three data sets. In a word, PCC does not fit for feature selection on high imbalance data sets.
- K-OFSD vs. ReliefF. K-OFSD gets higher predictive accuracy than ReliefF on all data sets with both KNN and SVM. ReliefF is similar to K-OFSD, because they both use the neighbors' information for feature selection. K-OFSD uses the **Card_Imbalanced** function for dependency calculation which makes it more suitable for class imbalance data. Thus, K-OFSD is superior to ReliefF on imbalance data.
- K-OFSD vs. MI. Both K-OFSD and MI perform stably. K-OFSD outperforms MI on all the seven data sets with both KNN and SVM. K-OFSD is a Rough Set based method which does not need to consider any domain knowledge other than the given dataset. This makes K-OFSD is more suitable than MI for class imbalance data sets.

In sum, K-OFSD provides the best overall performance in five of the seven data sets, while it is also comparable to the best competing approaches on the remaining two data sets. Meanwhile, K-OFSD gets the highest mean predictive accuracy with both KNN and SVM.

Table 6
G-MEAN predictive accuracy using KNN as the base classifier.

Data set	K-OFSD	Alpha-investing	OSFS	Fast-OSFS	SAOLA
DLBCL	**0.954**	0.4945	0.8991	0.9085	0.9235
LUNG	**0.9885**	0.9054	0.9495	0.9789	0.9761
LYM	**0.9467**	0.8639	0.8965	0.9050	0.9420
GLIOMA	**0.856**	0.804	0.8171	0.8181	0.8449
SRBCT	**1**	0.9669	0.9338	0.9316	0.9894
LUNG2	0.9646	**0.9795**	0.9447	0.9260	0.9468
CAR	**0.9615**	0.9449	0.9444	0.9449	0.9449
AVERAGE	**0.953**	0.851	0.912	0.916	0.938

Table 7
G-MEAN predictive accuracy using SVM as the base classifier.

Data set	K-OFSD	Alpha-investing	OSFS	Fast-OSFS	SAOLA
DLBCL	**0.9472**	0.2394	0.8176	0.9085	0.9358
LUNG	**0.9823**	0.9332	0.9600	0.9789	0.9795
LYM	0.9350	0.3474	0.8127	0.9050	**0.9579**
GLIOMA	**0.8754**	0	0.7214	0.8181	0.8667
SRBCT	**1**	0.9789	0.9113	0.9316	**1**
LUNG2	**0.9779**	0.9452	0.9656	0.9260	0.9394
CAR	**0.9615**	0.9449	0.9444	0.9449	0.9449
AVERAGE	**0.954**	0.627	0.876	0.916	0.946

Table 8
Running time (seconds).

Data set	K-OFSD	Alpha-investing	OSFS	Fast-OSFS	SAOLA
DLBCL	1.7052	**0.6171**	1.3934	0.9183	1.1944
LUNG	9.2873	1.2159	16.1068	**0.9714**	5.1406
LYM	1.1181	**0.1851**	1.0629	0.8916	0.9296
GLIOMA	1.2012	**0.2051**	0.9888	0.8201	0.8402
SRBCT	0.8133	**0.129**	0.7411	0.9421	0.5108
LUNG2	4.7875	**0.2449**	4.1598	0.9506	1.0128
CAR	12.4132	1.4197	4.8967	**0.9443**	2.073
AVERAGE	4.5	**0.5**	4.2	0.9	1.7

Table 9
The mean number of selected features.

Data Set	K-OFSD	Alpha-investing	OSFS	Fast-OSFS	SAOLA
DLBCL	10	**1.4**	1.7	3.5	17.1
LUNG	21.6	9.2	**2.7**	5	39.2
LYM	6.3	**1.2**	1.5	3.3	16.4
GLIOMA	4.1	**1**	1.3	2.7	9.7
SRBCT	5.1	13.2	**2.3**	3.9	17
LUNG2	16.5	12.2	**3**	5.3	19.5
CAR	**2**	31.1	3.2	4.2	19.8
AVERAGE	9	10	**2**	4	20

4.5. K-OFSD vs. online feature selection methods

In this section, we compare our algorithm with four state-of-the-art online feature selection methods: Alpha-investing [20], OSFS [15], Fast-OSFS [15], SAOLA [18].

All aforementioned algorithms are implemented in MATLAB [58]. The significance level α is set to 0.01 for OSFS, Fast-OSFS and SAOLA. For Alpha-investing, the parameters are set to the values used in [20]. We set $K = 7$ for K-OFSD as discussed in Section 4.2. We evaluate K-OFSD and the competing ones on G-MEAN predictive accuracy, number of selected features and running time.

Tables 6 and 7 summarize the G-MEAN predictive accuracy of K-OFSD against the other four algorithms using the KNN (k=1) and SVM classifiers. Tables 8 and Tables 9 show the running time and mean number of selected features of K-OFSD against other four algorithms.

From Tables 6 - 9 , we have the following observations.

- K-OFSD vs. Alpha-investing. In Table 8, Alpha-investing spends the least time in five of the seven data sets. Although Alpha-

investing is faster than K-OFSD on running time, it can not handle some data sets well. More specifically, in Tables 6 and 7, K-OFSD outperforms Alpha-investing on six of the seven data sets with both KNN and SVM. In addition, the features selected by Alpha-investing do not fit for the SVM classifier. For example, Alpha-investing with SVM only gets the predictive accuracy of 0, 0.2, and 0.3 on GLIOMA, DLBCL and LYM respectively. In Table 9, Alpha-investing only selects 1.4, 1.2 and 1 features on DLBCL, LYM and GLIOMA. The reason is that these data sets are very sparse and Alpha-investing can only select the first one or two features of these data sets.

- K-OFSD vs. OSFS. K-OFSD outperforms OSFS on all data sets with both KNN and SVM. On running time, K-OFSD is comparable to OSFS. In Table 9, we can see that OSFS selects the least mean number of features on these data sets. Thus, some important information is missing which causes the lower predictive accuracy.
- K-OFSD vs. Fast-OSFS. Fast-OSFS is faster than K-OFSD on the whole, but K-OFSD performs better than Fast-OSFS on all data sets with both KNN and SVM. Fast-OSFS also selects a very few number of features on data sets and this lead to missing some important information.
- K-OFSD vs. SAOLA. SAOLA is faster than K-OFSD. However, K-OFSD can beat SAOLA on all the seven data sets with KNN and can beat SAOLA on six of the seven data sets with SVM. Meanwhile, K-OFSD selects a less number of features than SAOLA while gets higher predictive accuracy. This demonstrates that the features selected by K-OFSD are more discriminative and less redundant.

In sum, our algorithm K-OFSD is not faster than some competing algorithms of Alpha-investing, SAOLA and Fast-OSFS, but it outperforms all competing algorithms with the highest mean predictive accuracy on these imbalance data sets.

5. Conclusions and future work

In this paper, we have formalized the problem of online streaming feature selection for class imbalance data and proposed an online feature selection framework OFSD based on dependency between features and classes. Unlike other online streaming feature selection approaches, OFSD selects features based on the dependency of either a single feature or a selected feature set to decision classes. According to the OFSD framework, a new algorithm K-OFSD was proposed for class imbalance data with feature streams in an online manner. In order to select features which can get high separability between the small class and large class, we use the information of a fixed number of neighboring instances near by the target object. As compared to five traditional imbalanced feature selection methods and four state-of-the-art online streaming feature selection algorithms, the proposed algorithm K-OFSD has demonstrated better performance for high-dimensional and class-imbalanced data. In our future work, we will improve the time performance of our algorithm by using more effective dependency calculation methods. Meanwhile, we will apply our algorithm to multi-class imbalanced data and other real-world applications with very high class imbalance data.

Acknowledgments

This work is supported in part by the National Key Research and Development Program of China under grant 2016YFB1000901, the Natural Science Foundation of China under grants (61503112, 61673152).

References

[1] H. Liu, H. Motoda, Computational Methods of Feature Selection, Chapman and Hall/CRC Press, 2007.
[2] M. Robnik-Sikonja, I. Kononenko, Theoretical and empirical analysis of relieff and rrelieff, Mach. Learn. 53 (1–2) (2003) 23–69.
[3] Q. Gu, Z. Li, J. Han, Generalized Fisher score for feature selection, in: Conference on Uai, 2011, pp. 266–273.
[4] J.R. Vergara, P.A. EstÉ¨vez, A review of feature selection methods based on mutual information, Neural Comput. Appl. 24 (1) (2014) 175–186.
[5] H. Peng, F. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1238.
[6] K. Benabdeslem, H. Elghazel, M. Hindawi, Ensemble constrained Laplacian score for efficient and robust semi-supervised feature selection, Knowl. Inf. Syst. 49 (3) (2016) 1161–1185.
[7] A. Alalga, K. Benabdeslem, N. Taleb, Soft-constrained Laplacian score for semi–supervised multi-label feature selection, Knowl. Inf. Syst. 47 (1) (2016) 75–98.
[8] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc. Ser. B(Methodol.) (1996) 267–288.
[9] V. Kumar, S. Minz, Multi-view ensemble learning: an optimal feature set partitioning for high-dimensional data classification, Knowl. Inf. Syst. 49 (1) (2016) 1–59.
[10] D. Wang, D. Irani, C. Pu, Evolutionary study of web spam: Webb spam corpus 2011 versus webb spam corpus 2006, in: Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures, in: CollaborateCom-2012, 2012, pp. 40–49.
[11] M. Wang, H. Li, D. Tao, K. Lu, X. Wu, Multimodal graph-based reranking for web image search, IEEE Trans. Image Process. 21 (11) (2012) 4649–4661.
[12] W. Ding, T.F. Stepinski, Y. Mu, L. Bandeira, R. Ricardo, Y. Wu, Z. Lu, T. Cao, X. Wu, Subkilometer crater discovery with boosting and transfer learning, Acm Trans. Intell. Syst. Technol. 2 (4) (2011) 1–22.
[13] X. Hu, P. Zhou, P. Li, J. Wang, X. Wu, A survey on online feature selection with streaming features, Front. Comput. Sci. (2016), doi:10.1007/s11704-016-5489-3.
[14] S. Hoi, J. Wang, P. Zhao, R. Jin, Online feature selection for mining big data, in: KDD BigMine 2012, 2012, pp. 93–100.
[15] X. Wu, K. Yu, W. Ding, H. Wang, X. Zhu, Online feature selection with streaming features, IEEE Trans. Pattern Anal. Mach. Intell. 35 (5) (2013) 1178–1192.
[16] J. Wang, M. Wang, P. Li, L. Liu, Z. Zhao, X. Hu, X. Wu, Online feature selection with group structure analysis, IEEE Trans. Knowl. Data Eng. 27 (11) (2015) 3029–3041.
[17] J. Wang, P. Zhao, S.C. Hoi, R. Jing, Online feature selection and its applications, IEEE Trans. Knowl. Data Eng. 26 (3) (2013) 698–710.
[18] K. Yu, X. Wu, W. Ding, J. Pei, Scalable and accurate online feature selection for big data, ACM Trans. Knowl. Discov. Data 11 (2) (2016) 16.
[19] S. Perkins, J. Theiler, Online feature selection using grafting, in: Proceedings of the 20th International Conference on Machine Learning, 2003, pp. 592–599.
[20] J. Zhou, D.P. Foster, R.A. Stine, L.H. Ungar, Streamwise feature selection, J. Mach. Learn. Res. 3 (2) (2006) 1532–4435.
[21] S. Eskandari, M. Javidi, Online streaming feature selection using rough sets, Int. J. Approx. Reason. 69 (C) (2016) 35–57.
[22] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, ACM SIGKDD Explor. Newsl. 6 (1) (2004) 1–6.
[23] S. Ando, Classifying imbalanced data in distance-based feature space, Knowl. Inf. Syst. 46 (3) (2016) 707–730.
[24] R. Pearson, G. Goney, J. Shwaber, Imbalanced clustering for microarray time-series, in: Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets, 2003.
[25] G. Wu, E.Y. Chang, Class-boundary alignment for imbalanced dataset learning, in: Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets, 2003, pp. 49–56.
[26] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[27] J.V. Hulse, T.M. Khoshgoftaar, A. Napolitano, R. Wald, Feature selection with high-dimensional imbalanced data, in: IEEE International Conference on Data Mining Workshops, 2009, pp. 507–514.
[28] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, ACM SIGKDD Explor. Newsl. 6 (1) (2004) 80–89.
[29] S. Maldonado, R. Weber, F. Famili, Feature selection for high-dimensional class-imbalanced data sets using support vector machines, Inf. Sci. (Ny) 286 (2014) 228–246.
[30] M. Wasikowski, X. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1388–1400.
[31] L. Yin, Y. Ge, K. Xiao, X. Wang, X. Quan, Feature selection for high-dimensional imbalanced data, Neurocomputing 105 (3) (2013) 3–11.
[32] Z. Pawlak, Rough Sets - Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Dordrecht , Boston, 1991.
[33] Y. Jing, T. Li, H. Fujita, Z. Yu, B. Wang, An incremental attribute reduction approach based on knowledge granularity with a multi-granulation view, Inf. Sci. (Ny) 411 (2017) 23–38.
[34] Y. Huang, T. Li, C. Luo, H. Fujita, S. jinn Horng, Dynamic variable precision rough set approach for probabilistic set-valued information systems, Inf. Sci. (Ny) 122 (2017) 131–147.
[35] J. Hu, T. Li, C. Luo, H. Fujita, S. Li, Incremental fuzzy probabilistic rough sets over two universes, Int. J. Approx. Reason. 81 (2017) 28–48.

[36] H. Chen, T. Li, Y. Cai, C. Luo, H. Fujita, Parallel attribute reduction in dominance-based neighborhood rough set, Inf. Sci. (Ny) 373 (2016) 351–368.
[37] L. T, G. Y, Computing on binary relations i: Data mining and neighborhood systems, in: Proceedings of the Rough Sets in Knowledge Discovery, 1998, pp. 107–121.
[38] Q. Hu, D. Yu, J. Liu, C. Wu, Neighborhood rough set based heterogeneous feature subset selection, Inf. Sci. (Ny) 178 (18) (2008) 3577–3594.
[39] Q. Hu, J. Liu, D. Yu, Mixed feature selection based on granulation and approximation, Knowl. Based Syst. 21 (4) (2008) 294–304.
[40] J. Zhang, T. Li, D. Ruan, D. Liu, Neighborhood rough sets for dynamic data mining, Int. J. Intell. Syst. 27 (4) (2012) 317–342.
[41] S.U. Kumar, H.H. Inbarani, Pso-based feature selection and neighborhood rough set-based classification for BCI multiclass motor imagery task, Neural Comput. Appl. (2016) 1–20.
[42] A. Shakiba, M.R. Hooshmandasl, Neighborhood system s-approximation spaces and applications, Knowl. Inf. Syst. 49 (2) (2016) 749–794.
[43] J. Mi, W. Wu, W. Zhang, Approaches to knowledge reduction based on variable precision rough set model, Inf. Sci. (Ny) 159 (3) (2004) 255–272.
[44] Z. Zheng, G. Wang, A rough set and rule tree based incremental knowledge acquisition algorithm, Fundam. Inform. 59 (2–3) (2004) 299–313.
[45] F. Hu, G. Wang, H. Huang, Y. Wu, Incremental attribute reduction based on elementary sets, in: Proceedings of 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC 2005), 2005, pp. 185–193.
[46] T. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowl. Based Syst. 20 (5) (2007) 485–494.
[47] D. Liu, T. Li, D. Ruan, W. Zou, An incremental approach for inducing knowledge from dynamic information systems, Fundam. Inform. 94 (2) (2009) 245–260.
[48] S. Li, T. Li, D. Liu, Dynamic maintenance of approximations in dominance-based rough set approach under the variation of the object set, Int. J. Intell. Syst. 28 (8) (2013) 729–751.
[49] D. Liu, T. Li, J. Zhang, Incremental updating approximations in probabilistic rough sets under the variation of attributes, Knowl. Based Syst. 73 (2015) 81–96.
[50] G. Lang, D. Miao, M. Cai, Three-way decision approaches to conflict analysis using decision-theoretic rough set theory, Inf. Sci. (Ny) 406–407 (2017) 185–207.
[51] G. Lang, D. Miao, T. Yang, M. Cai, Knowledge reduction of dynamic covering decision information systems when varying covering cardinalities, Inf. Sci. (Ny) 346–347 (2016) 236–260.
[52] Q. Hu, D. Yu, Z. Xie, Numerical attribute reduction based on neighborhood granulation and rough approximation, J. Softw. 19 (3) (2008) 640–649.
[53] P. Maji, S. Paul, Rough set based maximum relevance-maximum significance criterion and gene selection from microarray data, Int. J. Approx. Reason. 52 (2011) 408–426.
[54] R. Jensen, Q. Shen, Computational intelligence and feature selection: rough and fuzzy approaches, IEEE Press Series on Computational Intelligence, JOHN WILEY & SONS, 2008.
[55] K. Yang, Z. Cai, J. Li, G. Lin, A stable gene selection in microarray data analysis, IEEE Symp. Bioinf. Bioeng. 7 (1) (2005) 3–10.
[56] L. Yu, C. Ding, S. Loscalzo, Stable feature selection via dense feature groups, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: 40, 2008, pp. 803–811.
[57] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: One-sided selection, in: Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 179–186.
[58] K. Yu, W. Ding, X. Wu, Lofs: library of online streaming feature selection, Knowl. Based Syst. 113 (2016) 1–3.