Pattern Analysis and Applications
https://doi.org/10.1007/s10044-020-00876-7

THEORETICAL ADVANCES

Incremental one-class classifier based on convex–concave hull

Javad Hamidzadeh[1] · **Mona Moradi**[1]

Received: 1 August 2018 / Accepted: 30 March 2020

Abstract
One subject that has been considered less is a binary classification on data streams with concept drifting in which only information of one class (target class) is available for learning. Well-known methods such as SVDD and convex hull have tried to find the enclosed boundary around target class, but their high complexity makes them unsuitable for large data sets and also online tasks. This paper presents a novel online one-class classifier adapted to the streaming data. Considering time complexity, an incremental convex–concave hull classification method, called ICCHC, is proposed which can significantly reduce the computational time and expand the target class boundary. Also, it can be adapted to the gradual concept drift. Evaluations have been conducted on seventeen real-world data sets by hold-out validation. Also, noise analysis has been carried out. The results of the experiments have been compared with the state-of-the-art methods, which show the superiority of ICCHC regarding the accuracy, precision, and recall metrics.

Keywords One-class classification · Data stream · Online learning · Convex hull · Convex–concave hull

1 Introduction

One of the main issues in machine learning is the classification of objects according to their features. Traditional classifiers work well where the objects of all classes are sufficient, but in some real-world applications, e.g., recommendation tasks [1, 2], novelty detection [3], document detection [4], streaming mining, time series analysis [5], and anomaly detection [6], there are objects of only one class. This kind of classification is known as one-class classification (OCC) [7]. In OCC, the available class is called the target class, while the other classes are called the non-target classes.

Tax [7] categorized OCC methods into three main groups: density methods, reconstruction methods, and boundary methods. Density methods work based on estimating the probability density function (PDF) of the training data. However, these methods generally require a large object size. Moreover, density methods assume that noise and outliers are uniformly distributed in areas; therefore, if both target and non-target data lie in a low-density area, they may ignore target data. Classic density-based methods include k nearest neighbors (kNN) [8], Gaussian mixture model (GMM) [9], and Parzen density estimation [9].

Reconstruction methods use prior knowledge about the training data structure. k-means [8], self-organizing map (SOM) [10], and principal component analysis (PCA) [8, 11, 12] are some of the typical methods in this category.

Boundary methods are popular strategies that are to enclose the training data by, e.g., a hyperplane or a hypersphere, constructed by support vector machine (SVM) and support vector data description (SVDD) [13–17], or the convex hull [11]. The performance of SVDD depends upon the compactness of the constructed hypersphere and the employed kernel function. SVDD is inefficient for classifying high-dimensional data. Also, it does not work well if large density variations exist among the target data. However, when the volume of the hypersphere is increased, the chance of accepting outliers will be increased.

The convex hull geometry structure has been considered as a powerful tool in classification by defining the boundary of the target class [18–21]. Nevertheless, due to its high computational complexity, the traditional convex hull is unsuitable to be used in high dimensions. Some implementations were proposed to deal with this problem [22–26].

✉ Javad Hamidzadeh
J_Hamidzadeh@sadjad.ac.ir

Mona Moradi
Monamoradi0@gmail.com

[1] Faculty of Computer Engineering and Information Technology, Sadjad University of Technology, Mashhad, Iran

Published online: 15 April 2020

As mentioned above, OCC pertains to a binary classification task, which only information of one class (target class) is ample for learning. Due to this characteristic, OCC is suitable for online learning. Online learning is described as learning in a streaming data context in which training is done in consecutive rounds. Data streaming applications can vary from critical sensor data applications such as traffic management and weather forecasting to real-time decision support in industrial applications or weblog analysis, telecommunication, emails [27]. In online learning, a classifier is updated based on sequential objects arrived [28]. When new data is added, the current classifier should be updated instantly. In this situation, the prediction ability of the classifier is improved gradually. Online learning is used where dynamic adaption to the new patterns in the data is necessary, e.g., in the real-time recommendation [29], fraud detection [30–32], and spam detection [33, 34]. Computational time improvement is the main issue in online learning. In recent years, various online learning algorithms have been proposed to address this issue [35–39].

Noise is a difficulty in data streams that may appear unexpectedly or periodically and is harmful to the classification problem and must be filtered out. Therefore, improving robustness to noise is important for stream classifiers [40].

Concept drift in data streams is another important issue in classification problems. In real-world applications, the underlying distribution of data is non-stationary; thus, previously valid models might lose their validity by the passage of time. Broadly speaking, concept drift can be categorized into four groups in terms of speed: (1) sudden/abrupt drift which refers to a sudden replacement of a concept with another one. (2) Gradual drift which occurs in a period of time when the input data is switched from one source to another one. As time passes, the probability of sampling from the first source decreases gradually, but the probability of sampling from the auxiliary one increases. Note that, since the start of the occurrence of gradual drift, the objects from the auxiliary source might be considered as random noise. (3) Incremental drift, which occurs when the concept comes from more than two sources and the probability of choosing each concept, changes over time. (4) Reoccurring drift, which refers to the reoccurrence of a concept after a time interval. A perfect concept drift detection method in data streams is sensitive to real changes and can detect the location of changes. This method is insensitive to noise and distinguishes all types of drift.

To tackle the presence of noise and gradual concept drift, this paper presents an incremental convex–concave hull method for modeling OCC for the data stream by extending fuzzy logic. As pointed out, the process of selecting the convex vertices is time-consuming, especially when the number of training objects is large, and the dimension of the training objects is high. The proposed method intends to solve these weaknesses of convex hull-based OCC methods. Slight studies on one-class data stream classification and also insignificant one-class classification methods with incremental learning are the motivations to design the proposed method.

To evaluate the effectiveness of the proposed method in noise exposure, noise analysis was carried out. Slight changes in the accuracy reduction in the classification indicate the high efficiency of the proposed method.

The rest of this paper is organized as follows. In Sects. 2 and 3, the related work and preliminaries are introduced, respectively. The detail description of the proposed method is given in Sect. 4. Experimental results using benchmark data sets can be found in Sect. 5. Finally, Sect. 6 contains conclusions and future work.

2 Related work

This section includes a brief review of existing well-established data stream classification techniques and also reviews boundary-based methods used for OCC. Aggarwal [41] categorized the main techniques for data stream classification into: rule-based methods [42–46], nearest neighbor methods [47–49], neural network classifiers [50, 51], and ensemble methods [52–55]. Aggarwal [41] indicated that decision trees are the other well-known methods for data stream classification. The earliest method for decision tree-based stream classification named very fast decision tree (VFDT) was proposed based on the principle of the Hoeffding tree [56]. However, this method was not designed for concept-drifting data streams. To resolve this problem, Hulten et al. [57] proposed a method referred to as CVFDT (concept-adopting very fast decision tree learner). Hashemi and Yang [58] proposed a flexible decision tree structure based on fuzzy logic, named FlexDT (flexible decision tree), for data stream classification in the presence of concept change, noise, and missing values. The other decision-tree-based approaches are [59–61].

Besides, support vector methods can be employed for data stream classification. The well-known techniques in this category include support vector data description (SVDD) where the minimum hypersphere containing all the data is computed in a multidimensional space, and support vector machine (SVM) [62–64]. These methods classify the data objects correctly by defining the decision boundary.

The geometrical structure of the convex hull [65] is another method that has been used to define the class boundary in both multiclass [66] and one-class classification tasks [67]. Using the convex hull in real applications is limited by the fact that its computation in a high-dimensional space has an extremely high cost. Zeng et al. [11] proposed a one-class classification based on the convex hull (OCCCH) for bearing fault detection. OCCCH used the nearest point algorithm to

construct a convex hull. OCCCH is not incremental and is used for batch learning applications. Many studies have been proposed in the context of SVM related algorithms to reduce the complexity of convex hull. Bennett and Bredensteiner [68] incorporated the convex hull into a geometrical interpretation of SVM. Takahashi and Kudo [69] used the convex hull as a maximum margin classifier. They approximated the vertices of the convex hull, where the support planes were presented by a set of reflexive support functions separating one class from the other ones. Pal and Hattacharya [70] proposed a self-evolving a two-layer neural network model for computing the approximate convex hull of a set of points in 3-D and spheres. The vertices of the convex hull were mapped with the neurons of the top layer. CHVS [71] is a fast convex hull vertex selection algorithm for online classification, which converts convex hull into a linear equation problem with low computational complexity. Casale et al. [72] proposed a new method for one-class classification based on the convex hull geometric structure. Their process created a family of convex hulls abled to fit the geometrical shape of the training objects. Other approaches proposed for one-class classification are [67, 73–75].

Data stream classification is a variant of incremental learning of classifiers. In [76], a one-class classifier based on SVM for text stream is introduced. Vague one-class learning (VOCL) [77] is an ensemble method for the data stream which assigns a proper weight value to the classifiers. One-class ensemble classifiers (OcEC) [78] use a modified version of DBSCAN algorithm for expanding the positive class. It also uses ROC-SVM [79] or S-EM [80] algorithm for extracting negative samples from unlabeled data. One-class learning and concept summarization for data streams is another method that works on the data stream [81]. This method was based on VOCL [77] idea for learning concept, and it used a Markov model and set feature-based clustering techniques for summarization. Uncertain one-class classifier (UOCC) [82] was proposed for classification on uncertain streams. UOCC had two parts for classification. In the first part, UOCC created a bound score for each sample by using the local kernel-density-based method. This bound score was combined with the learning phase in a one-class classifier based on the SVM approach, and the second phase used the SVM clustering approach to summarize the concept from the history chunks. Online one-class SVM with active-set quadratic programming [83] is another approach for one-class classification on the data stream. In this method, an active-set method for updating the classifier model dealing with quadratic programming was proposed.

Incremental learning is a method of machine learning, in which input data is continuously used to extend the current model's knowledge. Many traditional machine learning algorithms inherently support incremental learning [84–86]. Most of these algorithms are based on changing SVM and SVDD from offline mode to the incremental mode. Wu et al. [87] proposed an improved SVDD algorithm, termed as self-adaptive SVDD (S-A-SVDD), which combined affinity propagation (AP) clustering algorithm and SVDD. Jiang et al. [88] proposed a second map support vector data description (SM-SVDD) method, which used an anomalous and close surface to realize analog circuit fault diagnosis. Cauwenberghs and Poggio [89] proposed a type of recursive SVM online learning. This method used incremental learning to solve classification recursively. It also used decremental "unlearning" as a forgetting approach. The main goal of this method was to keep Kuhn–Tucker (KT) conditions for all previously observed data. Krawczyk and Woźniak [90] introduced a one-class classifier with incremental learning and forgetting for data streams with concept drift. This classifier was based on WOCSVM (weighted one-class SVM), which proposed in [91]. Das et al. [92] proposed an SVM-based classification method for error detection in smart homes. An incremental weighted one-class classifier for mining stationary data stream, proposed by Krawczyk and Woźniak [93], was designed based on SVDD. This algorithm assigned weights to data points according to their distance from the center of the hypersphere.

Most of the online learning tasks can be reformulated as online convex optimization ones. Zhou et al. [94] presented an incremental convex hull algorithm based on online support vector regression in which SVM-kernel was used to update the convex hull boundary. Wang et al. [95] proposed an online SVM classifier by selecting convex hull vertices within each class.

The method presented in the present paper is different from the methods proposed in [94, 95]. The proposed method utilizes the convex optimization theory to improve the boundary constituted by SVDD. Also, it is capable of addressing concept drifting and detecting noise.

3 Preliminaries

In this section, the preliminary theories of support vector data description, convex hull, and fuzzy rough set are presented.

3.1 Support vector data description

Support vector data description (SVDD) [96] is based on the support vector machine (SVM). SVDD tries to create a hypersphere around the target class. Data inside the hypersphereare regarded as a target class, and data outside the hypersphereare regarded as non-target. The data objects that lie on the surface of the hypersphere named support vectors describe the boundary [97]. Figure 1 illustrates the example of the SVDD boundary in two-dimensional space.

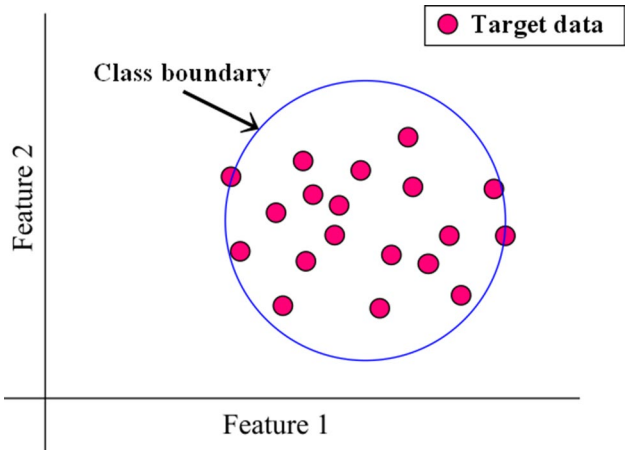

Fig. 1 The class boundary that SVDD detected

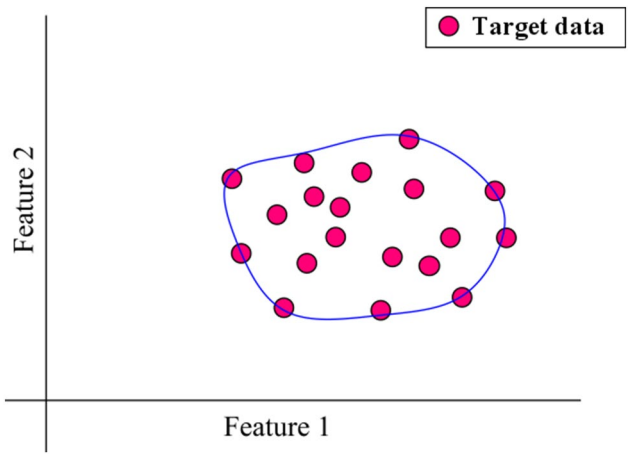

Fig. 2 SVDD boundary obtained by implementing kernel trick

Definition 1 SVDD defines a hypersphere; this sphere has two main parameters; one is the center of the sphere (a), and the other one is the radius (R), an object $(x_i)_{i=1}^n$ has three positions relative to the sphere as Eq. (1).

$$\begin{cases} inside & if \quad x_i - a^2 \leq R^2 \\ onsurface & if \quad x_i - a^2 = R^2 \\ outside & if \quad x_i - a^2 > R^2 \end{cases} \tag{1}$$

The principal goal of the SVDD is to find the hypersphere with minimum volume to contain all the data objects.

Definition 2 $F(R, a)$ is a minimization problem about the volume of the sphere that is presented in Eq. (2).

$$\begin{aligned} &\min F(R, a, \xi) = R^2 + C \sum_{i=1}^{N} \xi_i; \\ &s.t. \left\| x_i - a \right\|^2 \leq R^2 + \xi_i, \xi_i \geq 0, i = 1, 2 \ldots, n \end{aligned} \tag{2}$$

where a is the center, R is the radius of the hypersphere, C is the regularization parameter, and ξ_i is the slack variable. Equation (2) is a quadratic problem that is used in SVDD [98].

Definition 3 SVDD boundary is a closed curve that is crossed from nearby objects together of $S = \{x_i\}_{i=1}^m$ where $m \leq n$ and x_i can satisfy $x_i - a^2 = R^2$, S is the support vector set.

As shown in Fig. 1, there are some gap areas between the SVDD boundary and boundary points. Therefore, SVDD uses a kernel trick to optimize the class boundary [99]. Figure 2 represents the SVDD boundary obtained by implementing the kernel trick.

3.2 Convex hull

The convex hull of a set X of points $X \subset R^d$ is the minimum convex polygon P such that $P \subset X$ and contains all the points of X [65].

Definition 4 For $X = \{x_i\}_{i=1}^n \subset R^d$, P is a linear combination of X that is defined as Eq. (3).

$$P = CH(X) = \left\{ \sum_{i=1}^{k} \lambda_i x_i | \sum_{i=1}^{k} \lambda_i = 1, 0 \leq \lambda_i \leq 1, k \leq n \right\} \tag{3}$$

where each $x_i = \{x_i^1, x_i^2, \ldots, x_i^d\}$ is a d-dimensional vector in the feature space, and λ_i is the coefficient of the linear combination. P has k elements $P = \{x_i\}_{i=1}^k$ known as vertices set. A regular convex hull with k vertices (when $k \to \infty$) will be converted to a circle (see Fig. 3). In this case, the interior angle between the pairs of the nearest neighbors of a regular convex hull with k vertices becomes $\theta = 180 - \frac{360}{k}$, then $\lim_{k \to \infty} \theta = 180$. The convex hull functionality is approximately similar to the one-class classifiers like SVDD. Figure 4 illustrates a comparison of boundaries generated by SVDD and a convex hull.

Convexity of a polygon is a measurable trait that can be utilized to the detection of the boundary that closely matches the shape (see Fig. 5).

3.3 Fuzzy rough set

Fuzzy rough set [100] is constructed from a combination of the fuzzy set [101] and rough set [102]. Rough set tries to divide the universe of discourse to the lower approximation, boundary, and negative sets. To improve the application of

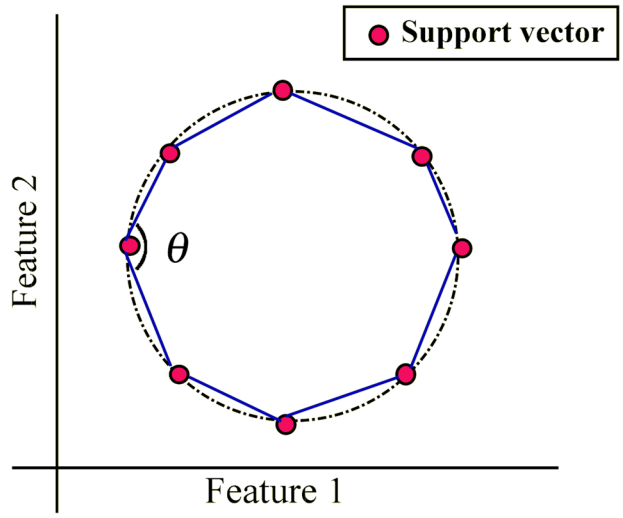

Fig. 3 A regular convex hull can be converted to a circle. The interior angle between a pair of the nearest neighbors is marked by θ

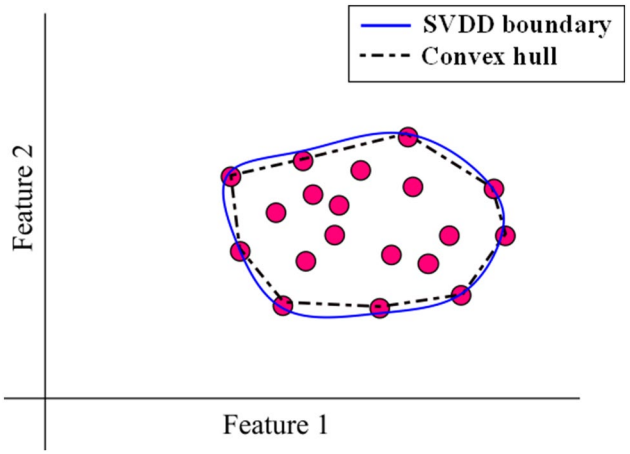

Fig. 4 A comparison of boundaries generated by SVDD (solid line) and a convex hull (dashed line)

rough set, Dubois and Prade [100] introduced a fuzzy version of this idea that can handle continuous attributes. Fuzzy rough set for each object returns a pair of memberships that show the lower approximation membership as a degree of certainty and upper approximation membership as a possibility degree of being included in target objects.

The lower and upper approximation memberships are constructed by indiscernibility relationships (IR) between objects and the amount of dependency on the target set, μ_F. IR as shown in Eq. (4), measures the similarity of each pair of objects. When two objects are identical, IR becomes 1, and when the objects are completely different, it shows zero.

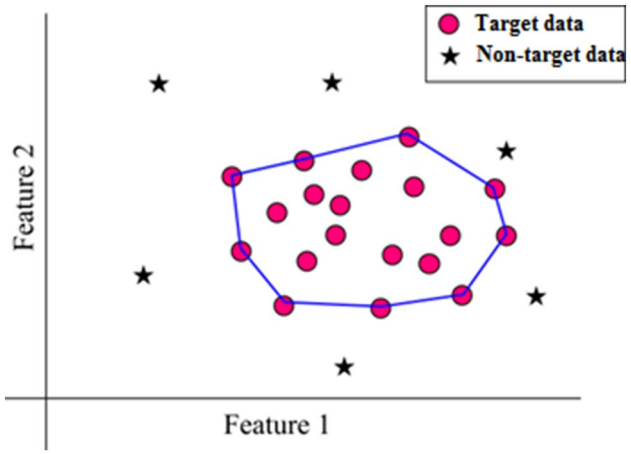

Fig. 5 Decision boundary generated by the convex hull to detect the target and the non-target data

$$IR(s_i, s_j) = \underset{a \in Dimension}{\tau} (1 - |s_i(a) - s_j(a)|^2) \tag{4}$$

According to the above equation, the differences between the two objects in every dimension are aggregated by τ which is a triangular fuzzy operator (t-norm). The result is called the indiscernibility relation between two objects under the decision boundary. On the other hand, membership of each object into the target class, μ_F, can be shown differently like binary ones or the others. Therefore, by these two concepts, the lower and upper approximation memberships are defined in Eqs. (5) and (6), respectively:

$$\mu_{\underline{F_{IR,\mu_F}}}(s_i) = \inf_{s_j \in T, s_i \neq s_j} \mathrm{I}\big(IR(s_i, s_j), \mu_F(s_j)\big) \tag{5}$$

$$\mu_{\overline{F_{IR,\mu_F}}}(s_i) = \sup_{s_j \in T, s_i \neq s_j} \tau\big(IR(s_i, s_j), \mu_F(s_j)\big) \tag{6}$$

where $\mu_{\underline{F_{IR,\mu_F}}}(s_i)\Big(\mu_{\overline{F_{IR,\mu_F}}}(s_i)\Big)$ is the degree of membership of s_i in $F_{\underline{IR,\mu F}}\Big(F_{\overline{IR,\mu F}}\Big)$ and the implicator (I) is calculated by $\max\big\{1 - IR(s_i, s_j), \mu_F(s_j)\big\}$. As shown in the above equations, two fuzzy operators, implicator (I) and t-norm (τ), combine two essential elements and result in several outputs. These fuzzy operators are generalized in [103] and are in the form of Lukasiewicz in this paper. Then, "*inf*" and "*sup*" select one of these outcomes as the final result. Hence, outliers and noise data can change the lower and upper approximation memberships in a wide range. Therefore, Verbiest et al. [104] proposed adjusted versions of the memberships using order weigh average (OWA) instead of "*inf*" and "*sup*". These forms of memberships are presented as follows:

$$\mu_{\underline{F_{IR,\mu_F}}}(s_i) = OWA_{\substack{\min \\ s_j \in T, s_i \neq s_j}} \mathrm{I}(IR(s_i, s_j), \mu_F(s_j)) \tag{7}$$

$$\mu_{\overline{F_{IR,\mu_F}}}(s_i) = OWA_{\substack{\max \\ s_j \in T, s_i \neq s_j}} \tau(IR(s_i, s_j), \mu_F(s_j)) \tag{8}$$

Consequently, the fuzzy rough set shows success in handling the vagueness and conflict among data [105, 106].

4 The proposed method

The proposed method focuses on designing an online OCC for data stream by using the convex–concave hull in which training data are learned incrementally. The proposed boundary-based classifier, named ICCHC (incremental convex–concave hull for one-class classification), is able to handle concept drift. The stages of ICCHC are demonstrated in Fig. 6.

4.1 Constituting the convex hull

When a new data object arrives, ICCHC constitutes the convex hull in three main steps: (1) the neighbors of the added data are found, (2) the new data position is set with regard to the convex hull, and (3) the convex hull is updated. Now, ICCHC is explained in detail.

1. *Finding the new data's neighbors* The idea of this step originated from Carathéodory's theorem [107]. According to this theorem, if X is a subset of a d-dimensional vector space, convex combinations of at most $d+1$ objects are sufficient. Therefore, the convex hull of a set X which contains three or more objects in the plane is the union of all the triangles determined by triples of objects from X, and more generally in d-dimensional space, the convex hull is the union of the simplices determined by at most $d+1$ vertices from X. In the training phase, as a new data x_{new} arrives, the closest vertex (called x_T) is found from the vertices of the convex hull.

Definition 5 $D(P, x_{new})$ is the distance function between $x_{new} \subset R^d$ and the convex hull P that is calculated by Eq. (9).

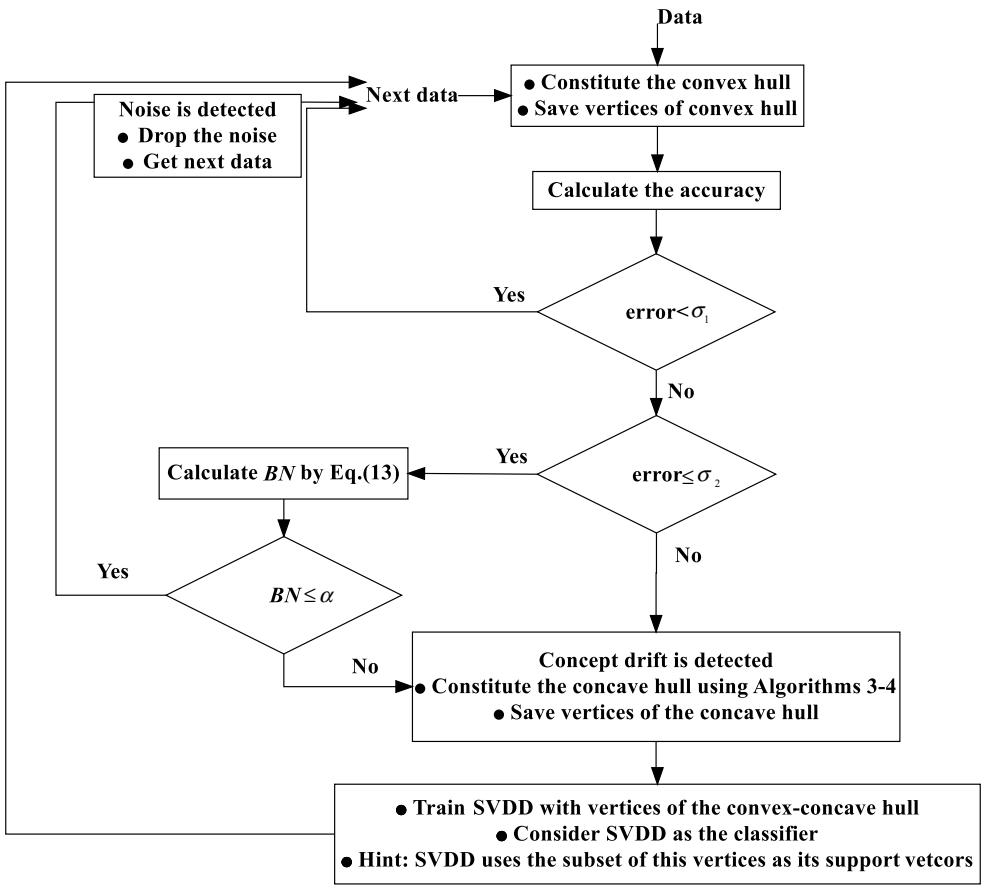

Fig. 6 ICCHC diagram

$$D^2(P, x_{new}) = \min (x_i - x_{new})^2 = \min \sum_{i=1}^{k} (\lambda_i x_i - x_{new})^2;$$
$$S.t. \sum_{i=1}^{k} \lambda_i = 1,\ 0 \le \lambda_i \le 1 \tag{9}$$

where $P = \{x_i\}_{i=1}^{k}$ is a convex set, k is the number of vertices of the convex hull, and λ is the coefficient of the linear combination. Next, $d-1$ neighbors from x_T should be added to $X' = \{x_{new}, x_T, x_1^{'}, x_2^{'}, \ldots, x_{d-1}^{'}\}^T$ where d is a maximum number of neighbors that are needed to be checked. X' will be used in constituting the optimum convex hull.

2. *Checking the position of the new data* The basic idea behind this step is to determine the position of the new data. The new data may place in three different situations than that of the convex hull (inside or outside the vertex).

Definition 6 Given $P = \{x_i\}_{i=1}^{k}$ is a convex set in R^d, consider the function $\Phi(x_{new})$ for every x_{new} object in R^d:

$$\sum_{i=1}^{k} \lambda_i x_i = x_{new}, \sum_{i=1}^{k} \lambda_i = 1, 0 \le \lambda_i \le 1 \tag{10}$$

$$\boldsymbol{\Phi}(x_{new}) = \sum_{i=1}^{k} \lambda_i x_i - x_{new} \tag{11}$$

For each x_{new}, the following cases hold:

1. If $0 \le \lambda_i < 1$, then $\Phi(x_{new}) \ge 0$. It is said that the x_{new} is in the interior of the convex hull.
2. If $\lambda_i > 1$, then $\Phi(x_{new}) < 0$. It is said that the x_{new} is on the exterior of the convex hull.

Then, Eq. (12) is used to specify the x_{new} position in a d-dimension convex hull.

$$\boldsymbol{\Phi}(x_{new}, P) = \begin{cases} inside & \boldsymbol{\Phi} \ge 0 \\ outside & \boldsymbol{\Phi} < 0 \end{cases} \tag{12}$$

In the case, if the new data lies inside the convex hull, this data is ignored, because it is unable to change the vertices of the convex hull, and on the contrary, if the new data lies outside the convex hull, it is added to the convex vertices set. Therefore, the convex hull is required to be updated, and some vertices may be removed from the convex set. Algorithm 1 presents this stage.

Algorithm 1 Checking the position of the newcomer.

Input: P (convex hull vertex set), x_{new} (new data object);
Output: P';
Compute $\Phi(x_{new}, P)$
if $\Phi >= 0$ **then**
  $P'=P$ // x_{new} is inside the convex hull and skip
else // x_{new} is outside the convex hull.
  $P' = P \cup \{x_{new}\}$ // x_{new} is added to the vertices of the convex hull.
  find x_T as the nearest vertex of P from the x_{new}
  d neighbors of x_T are regarded as x_i' in X'
  return Update convex hull (P, X')
end if
return P'

3. *Updating the convex hull* According to x_{new} obtained, the convex hull is updated. As mentioned before, when x_{new} is inside the convex hull, it cannot change the vertices of the convex hull. So, this object is ignored. In the case that x_{new} is outside the convex hull, it will be added to the convex vertices set. Hence, the convex hull is required to be updated. In this stage, a temporary convex hull is created with x_{new} and d selected vertices from $P' = X' \cup \{x_T\}$ set. The P' has $d+1$ elements. Algorithm 2 illustrates the updating phase.

Algorithm 2 Updating the convex hull.

Input: P (convex hull vertex set), $X'(x_{new}$ and d neighbors from it);
Output: P'

```
for i= 1 to d+1
    Compute φ(x'_i, P)
    if Φ >= 0 then            // x'_i is inside the new convex hull
        remove x'_i from P
    else                      //x'_i is outside the new convex hull
        remain x'_i in P
    end if
end for
return P'
```

Illustrative examples (Figs. 7, 8) demonstrate the proposed method. Figure 7 shows how the convex hull is updated without any deletion. As shown in Fig. 7a, there is a convex hull with four vertices $\{x_1, x_2, x_3, x_4\}$. The new object x_{new} wants to be added to the current convex hull, but it is outside the convex hull. First, the closest vertex to x_{new} should be found. It is x_1 in this example. Then, one other neighbor to x_1 is searched. Whereas both x_2 and x_3 are at the same distance to x_1, they are selected as two neighbors (see Fig. 7b). These vertices are added to $X' = \{x_{new}, x'_1, x'_2, x'_3\}$. The new object x_{new} and its two neighbors are selected and create a new temporary convex hull. Then, the other vertices in X' are checked to find out whether they are inside the new temporary convex hull or not, (see Figs. 7c–e). If one of the neighbors was in the convex hull, it must be removed from the main convex hull. Otherwise, the new data will be added to the vertices of the convex hull (see Fig. 7f).

Figure 8 is another example that shows how the convex hull is updated with deletion. As shown in Fig. 8a, there is a convex hull with four vertices $\{x_1, x_2, x_3, x_4\}$, and x_{new} will be added to the convex hull. The closest vertex to x_{new} is x_1 (see Fig. 8b). x_{new} and two desired neighbors of x_1 create a convex hull (see Fig. 8c–e). In Fig. 8e, x_1 is placed inside the convex hull, so it will be removed, and the main convex hull will be updated.

At the end of this stage, the primary classifieris attained. The decision boundary of ICCHC is different from SVDD boundary. According to the data distribution of the data set, ICCHC constitutes boundary for both convex sets (see Fig. 9a), and concave sets (see Fig. 9b).

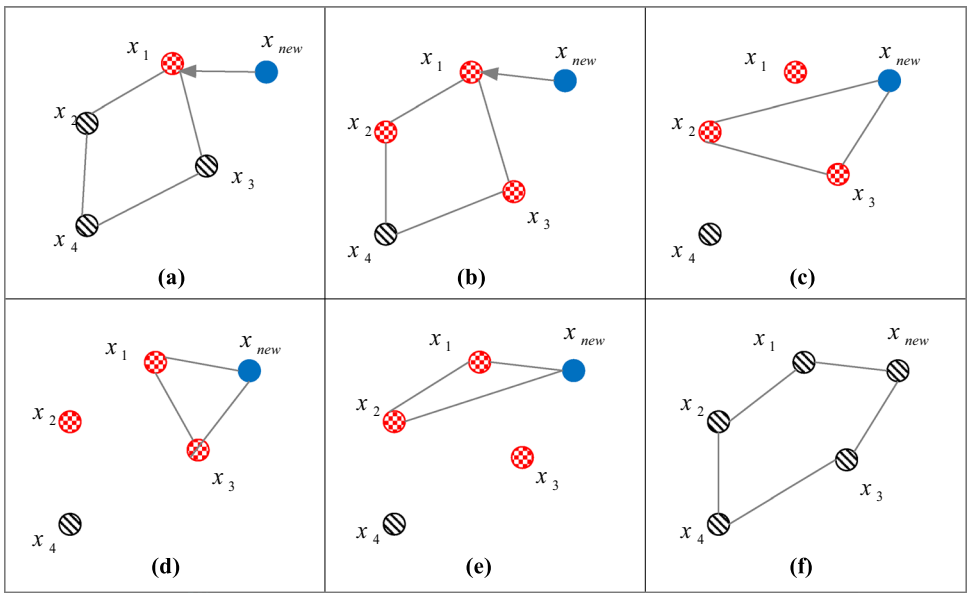

Fig. 7 An example of the updating convex hull without removing old vertex

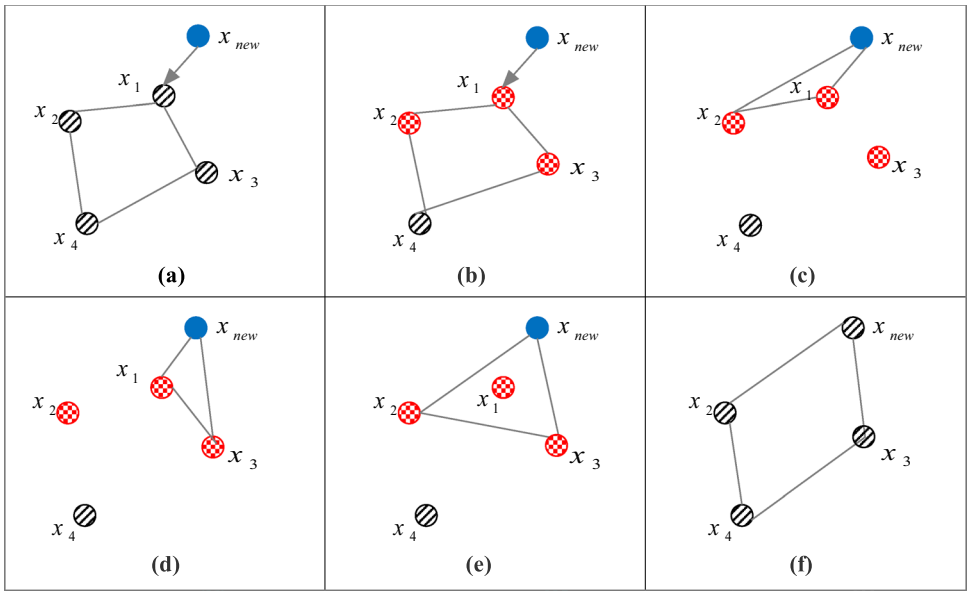

Fig. 8 An example of the updating convex hull with removing old vertex

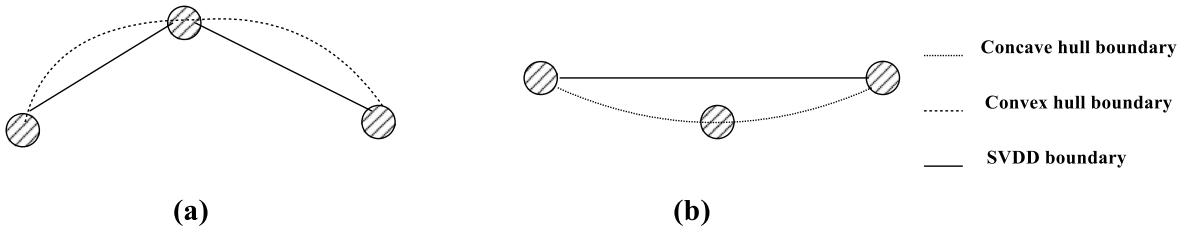

Fig. 9 Difference between the decision boundary in convex and concave states

4.2 Discovering the changes

For pattern mining, the sliding window technique is performed. In this technique, a window that satisfies the problem constraints is considered. All data streams in the current window W_c are utilized for data mining. For each mining round, the data streams of the current window W_c are stored in *buff_cur*. ICCHC finds the changes (noise and concept drift) by computing the difference between the lower and upper approximation membership of sliding windows.

The proposed method for the change detection comprises of three phases: (1) steady phase, (2) alarm phase, and (3) detection phase. More specifically, these are explained:

1. *Steady phase* This phase occurs when the classification error, that is 1-classification accuracy, is lower than the threshold σ_1. σ_1 is a predefined value which displays the maximum acceptable dissimilarity for remaining in the steady phase.
2. *Alarm phase* This phase occurs when the classification error is between two thresholds, σ_1 and σ_2. σ_2 is a predefined value that determines the minimum dissimilarity for detecting concept drift. During this phase, data reveal slight changes. ICCHC uses an appropriate strategy to distinguish between noise and concept drift. It utilizes the fuzzy rough set approach to diagnose noise and concept drift. The boundary region *BN*, given by Eq. (13), is considered as a feasible decision.

$$\begin{aligned} BN &= \mu_{\overline{F_{IR,\mu_F}}}(s_i) - \mu_{\underline{F_{IR,\mu_F}}}(s_i) \\ &= OWA_{\max\limits_{s_j \in buff\_cur, s_i \neq s_j}} \tau(IR(s_i, s_j), \mu_F(s_j)) - OWA_{\min\limits_{s_j \in buff\_cur, s_i \neq s_j}} \tau(IR(s_i, s_j), \mu_F(s_j)) \end{aligned} \tag{13}$$

where s is the data, and *buff_cur* is the set that contains the data. If the size of the boundary region becomes less than or equal to the adjustable parameter α, $0 \leq \alpha \leq 1$, it reveals that this uncertainty is due to the noise, on the contrary, if the size of the boundary region becomes higher than α, the concept drift is suspected.

In the case that noise is detected, it is ignored, and no change appears within the boundaries. Passing the time and arriving a new data, the old data must be ignored because of (1) expiration of the old data and (2) limitation in the

storage resource. In the case that concept drift is suspected, the convex hull cannot classify the objects accurately. So, the process of generating boundaries should be changed. This process is introduced in Sect. 4.4.

Note that, the greater α causing the more strict decision making, means noise is considered, and vice versa. The lower α, closer to 0, causing the soft decision making, means concept drift is considered.

3. *Detection phase* If the classification error is greater than the threshold σ_2, concept drift is detected. As mentioned earlier, the convex hull cannot classify the objects accurately. Hence, the process of generating boundaries should be changed. This process is introduced in Sect. 4.4.

4.3 Evaluating the classification accuracy

Regarding the boundary obtained from the previous stages, the classification accuracy is calculated (see Eq. (42)). By utilizing the accuracy obtained, the classification error is found (*classification error* = 1-*classification accuracy*). If the classification error is lower than the desired threshold, the convex hull boundary can separate the target class correctly. Otherwise, the boundary should be regenerated (see Sect. 4.4) to increase the classifier performance.

4.4 Constituting concave hull

To avoid drastic efficiency reduction, the vertices that are involved in determining the convex hull boundary should be changed. Therefore, both the vertices of a convex hull and the objects that are closed to the convex hull boundaries contribute to generating the vertices of the convex–concave hull *CCH(x)*.

$$CCH(X) = P \cup Close_{X_{Wc}} \tag{14}$$

where P is the vertices set of the convex hull and $Close_{X_{Wc}}$ is the set of objects in the current window W_c that are closed to the convex hull boundariesmeans that the members of $Close_{X_{Wc}}$ are chosen from stored streams in *buff_cur*.

This stage begins with selecting the most recent vertex x_i in P, continues by searching the closest objects to the old convex hull boundary (edge $\overrightarrow{x_i x_j}$), and terminates when x_i reaches to the adjacent vertex x_j. To find the closest objects to the boundary, k nearest neighbors are selected for each object. These objects are recorded in neighbors set. Each neighbor that has minimum angular distance with old edge $\overrightarrow{x_i x_j}$ is selected as a member of the $Close_{X_{Wc}}$ set.

$$\theta_{min} = \min_k \left[\cos^{-1} \left(\frac{\overrightarrow{x_i x_k} . \overrightarrow{x_k x_j}}{\left\| \overrightarrow{x_i x_k} \right\| . \left\| \overrightarrow{x_k x_j} \right\|} \right) \right] \tag{15}$$

Algorithm 3 Searching for vertices of a convex-concave hull.

Input: X (set of objects), k(number of neighbors)
Output: $CCH(X)$: convex-concave hull vertices

```
CCH(X) =∅
compute P (Apply algorithms 1 and 2)
for each pair of adjacent vertices (x_i, x_j) do
    calculate θ (using Eq. (15))
    apply algorithm 4 to search nearest objects Close_Xwc
    CCH(X) =CCH(X)∪ Close_Xwc
end for
```

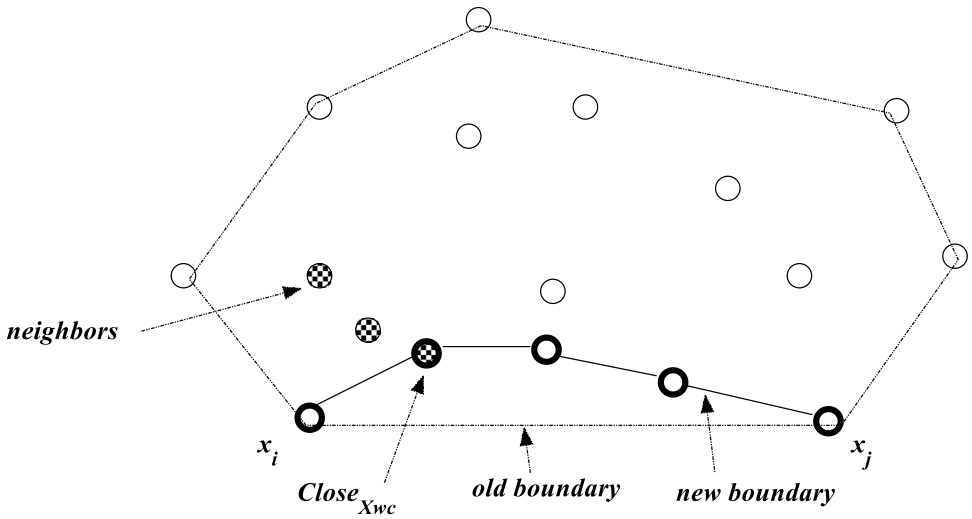

Fig. 10 Boundary regeneration. In this figure, 3-nearest neighbors are selected as neighbors

Algorithm 4 Searching for the nearest objects $Close_{Xw_c}$.

Input: X (set of objects), k (number of neighbors), θ (the angle between adjacent vertices), $\{x_i, x_j\}$ (two adjacent vertices of P)

Output: $Close_{Xw_c}$: The set of objects that are closed to the edge defined by adjacent vertices $\{x_i, x_j\}$

beginning $=x_i$
ending $=x_j$
$Close_{Xw_c}=x_i$
  while beginning $\neq x_j$ **do**
    neighbors=$\{x_1, \ldots, x_k\}$ //get k nearest objects to beginning
      for each object $x_i \in$ neighbors
        Define *a new boundary* as a new edge for vertices x_i and beginning;
        if *new boundary* does not intersect the convex-concave hull **then**
          $Close_{Xw_c}=Close_{Xw_c} \cup x_i$
        end if
      end for
    if no object was added in the last loop **then**
      $Close_{Xw_c} = Close_{Xw_c} \cup x_j$;
      Return $Close_{Xw_c}$;
    end if
  remove beginning from X;
  update θ using the two more recently added objects of $Close_{Xw_c}$;
    if beginning = x_j **then**
      Return $Close_{Xw_c}$;
    end if
  beginning $=x_i$;
  end while

The concept of constituting concave hull is demonstrated in Fig. 10. In each iteration, 3-nearest neighbors are selected as neighbors. The closest objects are found by Algorithms 3 and 4.Eventually, the new boundary is generated.

4.5 Creating a soft boundary

To increase the classification accuracy, creating a soft boundary and seeking the smallest hypersphere should be done. To this purpose, the SVDD approach is used. This stage is made up of using convex-concave vertices as training data. Assuming there are objects $CCH(X) = \{x_i\}_{i=1}^{m}$ which can be figured by center a, and radial R to cover all the x_i. The hypersphere is formulated as Eq. (16).

$$\min F(R, a, \xi_i) = R^2 + C\sum_{i=1}^{m}\xi_i;$$
$$s.t. \|x_i - a\|^2 \leq R^2 + \xi_i, \xi_i \geq 0, i = 1, 2 \ldots, m \quad (16)$$

where C is the regularization parameter which controls the trade-off between the volume of hypersphere and the training error caused by allowing outliers in the class description, and ξ_i are slack variables. Increasing the value of C, causing more training objects to fall outside the class boundary. Then, Lagrange-based optimization is applied to find the optimal parameter values. The Lagrangian function is given by Eq. (16):

$$L(R, a, \xi_i, \alpha_i, \gamma_i) = R^2 + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\gamma_i\xi_i - \sum_{i=1}^{m}\alpha_i[R^2 + \xi_i - (x_i.x_j - 2a.x_i + a.a)]$$
$$s.t.\ \alpha_i > 0;\ \gamma_i > 0 \quad (17)$$

where α and γ are Lagrange multipliers, and $x_i.x_j$ is an inner product. For each data object x_i, there is only one pair of corresponding Lagrange modulus α_i and γ_i, to meet it. L should be minimized with respect to R, a, ξ and maximized with respect to α_i and γ_i. Setting partial derivatives to zero gives the constraints:

$$\frac{\partial L}{\partial R} = 0 \Rightarrow \sum_i \alpha_i = 1 \tag{18}$$

$$\frac{\partial L}{\partial a} = 0 \Rightarrow a = \frac{\sum_i \alpha_i x_i}{\sum_i \alpha_i} = \sum_i \alpha_i x_i \tag{19}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow C - \alpha_i - \gamma_i = 0 \tag{20}$$

Resubstituting Eqs. (18)–(20) into Eq. (17) results in:

$$L(R, a, \xi, \alpha, \gamma) = \sum_{i=1}^{m} \alpha_i (x_i . x_j) - \sum_{i=1, j=1}^{m} \alpha_i \alpha_j (x_i . x_j) \tag{21}$$

Notice that there is a Lagrange multiplier α_i for every training object. In the solution, those objects for which $\alpha_i > 0$ are called *support vectors*, and lie on the hypersphere. The radial of a hypersphere (R) can be calculated by using Eq. (22).

$$R^2 = (x_k . x_k) - 2 \sum_{i=1}^{m} \alpha_i (x_i . x_k) + \sum_{i=1, j=1}^{m} \alpha_i \alpha_j (x_i . x_j) \tag{22}$$

Once the new data x_{new} is arrived, Eq. (23) is used to judge whether it belongs to the target class or not.

$$\|x_{new} - a\|^2 = (x_{new} - x_{new}) - 2 \sum_{i=1}^{m} \alpha_i (x_{new} . x_i) + \sum_{i=1, j=1}^{m} \alpha_i \alpha_j (x_i . x_j) \leq R^2 \tag{23}$$

If Eq. (23) holds, it can be concluded that the new data x_{new} belongs to the target class. Otherwise, it belongs to a non-target class. Now, to handle non-linearly separable problems, the kernel method is used. The main idea of the kernel method is to utilize a kernel function $K(x_i . x_j)$ instead of $x_i . x_j$ to map the non-linear problem of low-dimension into the linear problem of high-dimension [108]. In this paper, the Radial Basis Function (RBF kernel) (see Eq. (24)) is adopted.

$$K(x_i, x_j) = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \tag{24}$$

4.6 Statistical stability of the proposed method

The main purpose of classification problems is estimating real data distribution function with minimum error. Hence, a mathematics tool is needed to determine the estimating error of the decision surface and also an upper bound of this error. To improve the generalization capability of the proposed method, the upper bound error of the proposed method is determined using that tool. It is required to use a metric for evaluating function complexity to improve the generalization capability. This metric must be dependent on the data distribution to obtain the near error bound in the classification problem. Hence, Radmacher complexity metric is used [109]. The excellence of this metric is its dependency on the data resulted in a bound near the real error. Therefore, an upper bound is achieved for generalization capability based on each problem data using Radmacher complexity metric. Hence, it is a tool for evaluating the performance of the proposed decision surface. The summary of notations used in this subsection is shown in Table 1.

4.6.1 The generalization capability of the proposed method

In this subsection, the generalization capability of the proposed method is proven.

Theorem 1 *Let D be a stationary probability distribution function which is unknown. Assume that* $Y = (Z, l)\, Z = (X, Y)$ *consists of real data Z and their labels represented by* $l = \{-1, +1\}$ $Y = \{-1, +1\}$. *If there is a subset with the size of n with D distribution, with probability at least* $1 - \delta$ *the upper bound of the expected error is defined as Eq. (25).*

$$\Gamma\left[lh_{ICCHC}(z) \leq 0\right] \leq\frown \Gamma_{n_i}\left[LC \times (lh_{ICCHC}(z))\right] + 2 \times LF \times R_{n_i}(H_{ICCHC}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2n_i}} \tag{25}$$

where LF and LC are function and Lipschitz constant, respectively. $h_{ICCHC}(z)$ *is the function of SVDD for checking*

Table 1 Summary of notations

Symbol	Definition
Z	Test object
z_i	Training object
z_i^j	Instance i from class j
N	Total number of training instances
n_i	Number of class i instances
$Loss_{h_{ICCHC}(y)}$	Loss value of function h_{ICCHC} for test instance y
H_{ICCHC}	Basic classifier function family
$R_n(H_{ICCHC})$	Radmacher metric in functions family H_{ICCHC}
$\hat{G}_n(H_{ICCHC})$	Empirical Gaussian metric in functions family H_{ICCHC}
$G_n(H_{ICCHC})$	Gaussian metric in functions family H_{ICCHC}
LF	Lipschitz function
LC	Lipschitz constant
ξ	Radmacher random variables
Γ	Real error of classifier
E_{z_i}	Expected value for training instances
$\hat{E}_s\left[L_{h_{ICCHC}(y)}\right]$	Empirical error of classifier
E_D	Expected value for D data distribution

whether the object z is in the hypersphere or not. $h_{ICCHC}(z)$ *is defined in Eq. (26).*

$$h_{ICCHC} = h(f(z)) = \begin{cases} -1 & if\ f(z) \le 0 \\ +1 & otherwise \end{cases} \tag{26}$$

where f(z) will be defined as follows:

$$dist^2(z) - R^2 = -2\sum_i \alpha_i^* k(z, x_i) + 2\sum_i \alpha_i^* k(x_{sv}, x_i) \tag{27}$$

where $dist^2(z)$ *is a square value of the distance between object z and the center of the hypersphere,* $\alpha_i^{*'}s$ *are the coefficients of support vectors, and* $x'_{sv}s$ *are the support vectors. Since the proposed decision surface is convex, the complexity of hybrid family functions of the proposed method is equal to the complexity of basic family. To complete the proof of theorem 1, Eq. (28) is given:*

$$\begin{aligned} R_{n_i}(\Phi \circ H_{ICCHC}) &= E_{z_i} E_\xi \left[\sup_{\varphi \circ h_{ICCHC} \in \Phi \circ H_{ICCHC}} \left| \frac{2}{n_i} \sum_{i=1}^{n_i} \xi_i \varphi(h_{ICCHC}(z_i)) \right| \right] \\ &\le 2 Loss_\xi E_{z_i} E_\xi \left[\sup_{h_{ICCHC} \in H_{ICCHC}} \left| \frac{2}{n_i} \sum_{i=1}^{n_i} \xi_i h_{ICCHC}(z_i) \right| \right] \\ &= 2 \times LC \times R_{n_i}(H_{ICCHC}) \end{aligned} \tag{28}$$

The third part of the Eq. (25) on the right side and also the proof completion of theorem 1 is achieved by theorem 2.

Theorem 2 [109] *Let D be a stationary probability distribution function which is unknown. Assume that* $Y = (Z, L), Z = (X, Y)$ *consists of real data Z and their labels represented by* $L = \{-1, +1\}$ $Y = \{-1, +1\}$. *If there are n independent data with D distribution, with probability at least* $1 - \delta$, *the upper bound error of the basic functions h is defined as Eq. (29):*

$$E_D\left[Loss_{h_{ICCHC}}(x)\right] \le \hat{E}_s\left[Loss_{h_{ICCHC}}(y)\right] + \frac{1}{2} R_{n_i}(H_{ICCHC}) + \sqrt{\frac{\ln(\frac{1}{\delta})}{2n_i}} \tag{29}$$

where $\hat{E}_s\left[Loss_{h_{ICCHC}}(y)\right]$ *is the training error.*

4.6.2 Upper bound on the expected value of the decision surface classification error

In this subsection, Radmacher complexity metric is used to determine the upper bound on the expected value of the decision surface classification error. This metric determines the worst bound which is the nearest bound to the real bound of error. This bound is dependent on the complexity of functions family and data distribution.

Lemma 1 [109] *The empirical Gaussian metric of function family* $h_{ICCHC}(x)$ *is:*

$$\hat{G}_{n_i}(h_{ICCHC}) \le \frac{2}{n_i} \left[\sum_{i=1}^{n_i} s(z_i, z_i) \right]^{\frac{1}{2}} \tag{30}$$

where empirical Gaussian metric and Gaussian metric are: (z is training instance)

$$\hat{G}_{n_i}(H_{ICCHC}) = E_k \left[\sup_{h_{ICCHC} \in H_{ICCHC}} \left| \frac{2}{n_i} \sum_{i=1}^{n_i} k_i h_{ICCHC}(z_i) \right| \right] \tag{31}$$

$$G_{n_i}(H_{ICCHC}) = E_{z_i} \hat{G}_{n_i}(H_{ICCHC}) \tag{32}$$

where the distribution of variables k_i *is standard normal.*

Proof The proposed functions family sets are:

$$\begin{aligned} H_{ICCHC} &= \left\{ \sum_{i=1}^{n_i} \lambda_i s(z, z_i) : n_i \in N, z_i \in Z \right\} \\ &\subseteq \left\{ < \lambda, \Phi(z) > : \|\lambda\|_1 \le 1 \right\} \end{aligned} \tag{33}$$

Based on the property of the empirical Gaussian complexity metric for a subset of functions and the property of sup function, Eq. (34) is obtained:

$$\hat{G}_{n_i}(H_{ICCHC}) \le E_k \left[\sup_{\|\lambda\|_1 \le 1} < \lambda, \frac{2}{n_i} \sum_{i=1}^{n_i} k_i \Phi(z_i) > \right] \tag{34}$$

Based on *Cauchy–Schwartz*, Eq. (35) is given:

$$\begin{aligned} \sup_{\|\lambda\|_1 \le 1} < \lambda, \sum_{i=1}^{n_i} k_i \Phi(z_i) > &= < \frac{\sum_{i=1}^{n_i} k_i \Phi(z_i)}{\left\| \sum_{i=1}^{n_i} k_i \Phi(z_i) \right\|}, \sum_{i=1}^{n_i} k_i \Phi(z_i) > \\ &= \frac{1}{\left\| \sum_{i=1}^{n_i} k_i \Phi(z_i) \right\|} < \sum_{i=1}^{n_i} k_i \Phi(z_i), \sum_{i=1}^{n_i} k_i \Phi(z_i) > = \left\| \sum_{i=1}^{n_i} k_i \Phi(z_i) \right\| \end{aligned} \tag{35}$$

Cauchy–Schwartz unequal is given in Eq. (36):

$$< \lambda, \sum_{i=1}^{n_i} k_i \Phi(z_i) > \le \|\lambda\| \left\| \sum_{i=1}^{n_i} k_i \Phi(z_i) \right\| \tag{36}$$

Equation (37) is obtained by Eqs. (34) and (35):

Table 2 Data sets characterizations

Data sets name	#of attributes	#of objects	Target class	#of target	#of others
Breast cancer (Wisconsin)	10	683	Class 2	444	239
			Class 4	239	444
Balance scale	4	625	Class 1	49	576
			Class 2	288	337
			Class 3	288	337
Glass identification	10	214	Class 1	70	144
			Class 2	76	138
			Class 3	17	197
			Class 5	13	201
			Class 6	9	205
			Class 7	29	185
Ionosphere	33	351	Class g	225	126
			Class b	126	225
Blood transfusion service center	5	748	Class 0	570	178
			Class 1	178	570
Iris	4	150	Class Se	50	100
			Class Ve	50	100
			Class Vi	50	100
Seeds	7	210	Class 1	70	140
			Class 2	70	140
			Class 3	70	140
Liver	6	345	Class 1	145	200
			Class 2	200	145
SPECTF heart	44	267	Class 0	212	55
			Class 1	55	212
Diabetes	8	768	Class 1	500	268
			Class 2	268	500
Segment	19	2310	Class 1	330	1980
			Class 2	330	1980
			Class 3	330	1980
			Class 4	330	1980
			Class 5	330	1980
			Class 6	330	1980
			Class 7	330	1980
Vehicle	18	946	Class 1	240	706
			Class 2	240	706
			Class 3	240	706
			Class 4	226	720
Musk	168	476	Class 0	207	269
			Class 1	269	207
Yeast	8	1484	Class CYT	463	1021
			Class NUC	429	1055
			Class MIT	244	1240
Letter recognition	16	20,000	Class A	789	19,211
			Class E	768	19,232
			Class I	755	19,245
			Class O	753	19,247
Nursery	8	12,960	Class N	4320	8640
			Class P	4266	8694
Chess	6	28,056	Class 13	4194	23,862
			Class 14	4553	23,503
			Class 15	2166	25,890

$$\hat{G}_{n_i}(H_{ICCHC}) \leq \frac{2}{n_i} E_k \left[\left\| \sum_{i=1}^{n_i} k_i \Phi(z_i) \right\| \right] = \frac{2}{n_i} E_k \left[\left(\sum_{i=1}^{n_i} \sum_{j=1}^{n_i} k_i k_j s(z_i, z_j) \right)^{\frac{1}{2}} \right] \tag{37}$$

Since the square root function is concave and according to the *Jensen* theorem:

$$\begin{aligned} \hat{G}_{n_i}(H_{ICCHC}) &\leq \frac{2}{n_i} E_k \left[\left(\sum_{i=1}^{n_i} \sum_{j=1}^{n_i} k_i k_j s(z_i, z_j) \right)^{\frac{1}{2}} \right] \\ &\leq \frac{2}{n_i} \left(\sum_{i=1}^{n_i} \sum_{j=1}^{n_i} E_k \left[k_i k_j s(z_i, z_j) \right] \right)^{\frac{1}{2}} \end{aligned} \tag{38}$$

Given that the empirical average of expected value is equal to the average community:

$$\begin{aligned} E_{z_i} \left(\frac{2}{n_i} \left(\sum_{i=1}^{n_i} s(z_i, z_i) \right)^{1/2} \right) &= 2 \left[\frac{1}{n_i} E_{z_i} \left(\sum_{i=1}^{n_i} \frac{s(z_i, z_i)}{n_i} \right) \right]^{1/2} \\ = 2 \left[\frac{1}{n_i} E_{z_i} s(Z, Z) \right]^{1/2} &= 2 \sqrt{\frac{E_{z_i} s(Z, Z)}{n_i}} \end{aligned} \tag{39}$$

□

5 Experimental results

In this section, the experimental results of the proposed method are reported. The used data sets are introduced in Sect. 5.1, concisely. The evaluation metrics are expressed in Sect. 5.2. Extensive experiments have been carried out to evaluate the proposed method against four other methods. These results and their interpretation are described in Sect. 5.3. Moreover, a detailed noise analysis in Sect. 5.4 is presented.

5.1 Data sets

To analyze the effectiveness of the proposed method, experiments were conducted on seventeen real data sets from the UCI machine learning repository (available at http://archive.ics.uci.edu/ml). Table 2 shows the characteristics of the data sets. For each data set listed in this table, one of the classes (referred to as the target class) is well-characterized by objects in the training data. All data sets were divided into chunks of objects in each, given sequentially to the classifier.

5.2 Evaluation metrics

The experiments have been performed in the MATLAB R2013 on an Intel processor at 2.30 GHz with 4 GB of RAM. Evaluation metrics such as precision, recall, accuracy, and computational time have been applied to compare the efficiency of the proposed method and four other methods. These estimators are determined by:

$$Precision = \frac{TP}{(TP + FP)} \tag{40}$$

$$\text{Re}call = \frac{TP}{TP + FN} \tag{41}$$

$$Accuracy = \frac{TP + TN}{(TP + FN) + (FP + TN)} \tag{42}$$

where *TP, TN, FP*, and *FN* are introduced in Table 3. Furthermore, the computational time is considered as the length of time required to perform a computational process.

5.3 Results and interpretation

As previously mentioned, the proposed one-class classifier intends to solve the weaknesses of convex hull-based methods by reducing the computational time. The primitive boundaries generated by the proposed method are hard. Depending on the data distribution and the efficiency of the classifier, it decides whether to perform smoothness or not. Hence, it does not require complicated computing.

The experimental results are expressed in Tables 4, 5, 6 and 7. These tables indicate the classification performance of ICCHC and four different boundary-based OCC algorithms: Conventional SVDD (C-SVDD) [96], Self-Adaptive SVDD (S-A-SVDD) [87], Online SVM based on VS (VS-OSVM) [95], Adaptive weighted one-class support vector machine (A-WOCSVM) [90], in terms of precision, recall, accuracy, and computational time. Furthermore, noise analysis on one class of each data set according to [110] was performed. In the implementation of competing methods, the well-known library LibSVM for SVDD [111] and the RBF kernel were used.

For all the experiments, data sets were divided using holdout validation, i.e., a subset of objects is chosen at random to form the test set, and the remaining observations are retained as the training set. In this paper, 10% of data were

Table 3 Confusion matrix

	Objects from the target class	Objects from the non-target class
Predicted as a target object	True positive (*TP*)	False positive (*FP*)
Predicted as a non-target object	False negative (*FN*)	True negative (*TN*)

Table 4 The percentage of precision and average rank

Data sets name	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
Breast cancer (Wisconsin)	Class 2	51.72(2)	50.85(3)	50.02(4)	47.83(5)	**88.89(1)**
	Class 4	51.61(4)	52.43(3)	50.63(5)	54.55(2)	**64.87(1)**
Balance scale	Class 1	44.05(5)	**55.45(1)**	50.17(3)	47.37(4)	54.84(2)
	Class 2	51.72(3)	52.61(2)	**61.92(1)**	42.11(5)	51.35(4)
	Class 3	51.65(3)	**53.05(1)**	50.05(4)	45.04(5)	51.85(2)
Glass identification	Class 1	55.56(3)	55.71(2)	**62.51(1)**	45.45(5)	54.55(4)
	Class 2	50.13(4)	55.25(3)	55.56(2)	44.41(5)	**73.79(1)**
	Class 3	50.07(4)	55.94(2)	55.57(3)	44.48(5)	**71.43(1)**
	Class 5	55.51(2)	50.77(3)	44.44(5)	50.18(4)	**63.64(1)**
	Class 6	50.10(5)	50.92(3)	54.55(2)	50.41(4)	**66.67(1)**
	Class 7	55.56(2)	54.71(3)	50.33(5)	50.62(4)	**72.73(1)**
Ionosphere	Class g	52.63(4)	53.79(3)	55.16(2)	50.51(5)	**56.52(1)**
	Class b	**53.33(1)**	47.53(4)	52.38(2)	50.15(3)	42.11(5)
Blood transfusion service center	Class 0	56.76(2)	52.05(4)	50.16(5)	53.13(3)	**67.39(1)**
	Class 1	54.84(4)	56.28(3)	52.14(5)	57.14(2)	**66.07(1)**
Iris	Class Se	49.56(4)	50.62(3)	57.04(2)	**57.14(1)**	44.44(5)
	Class Ve	49.85(4)	50.48(3)	49.16(5)	54.55(2)	**57.14(1)**
	Class Vi	**50.01(1)**	49.92(2)	49.61(4)	49.46(5)	49.83(3)
Seeds	Class 1	50.20(3)	50.89(2)	49.97(4)	**57.14(1)**	45.45(5)
	Class 2	50.07(5)	**56.45(1)**	54.13(4)	55.56(2)	54.55(3)
	Class 3	**55.56(1)**	54.85(2)	49.01(3)	41.67(5)	46.15(4)
Liver	Class 1	52.38(3)	45.72(5)	52.63(2)	49.04(4)	**69.74(1)**
	Class 2	52.63(2)	50.44(4)	52.17(3)	49.29(5)	**57.89(1)**
SPECTF heart	Class 0	53.01(3)	**53.82(1)**	53.13(2)	50.04(5)	51.84(4)
	Class 1	**55.56(1)**	54.66(2)	50.46(3)	50.13(4)	46.15(5)
Diabetes	Class 1	50.75(3)	47.15(4)	51.72(2)	32.43(5)	**59.62(1)**
	Class 2	54.29(2)	**56.86(1)**	51.35(4)	38.24(5)	52.73(3)
Segment	Class 1	48.25(4)	**53.44(1)**	53.13(2)	42.86(5)	50.36(3)
	Class 2	**54.46(1)**	53.20(3)	51.92(4)	45.19(5)	54.11(2)
	Class 3	53.64(3)	54.75(2)	50.19(4)	43.53(5)	**59.59(1)**
	Class 4	48.54(4)	53.29(2)	50.55(3)	37.96(5)	**61.27(1)**
Vehicle	Class 1	50.17(3)	50.20(2)	**53.19(1)**	48.84(4)	48.10(5)
	Class 2	52.73(2)	51.16(3)	**53.06(1)**	44.31(5)	50.94(4)
	Class 3	50.14(4)	51.81(3)	54.17(2)	45.65(5)	**58.70(1)**
	Class 4	51.85(4)	53.13(3)	53.66(2)	50.12(5)	**64.58(1)**
Musk	Class 0	53.85(3)	52.02(4)	**61.54(1)**	44.23(5)	57.14(2)
	Class 1	**57.69(1)**	48.44(5)	51.42(4)	51.72(3)	55.17(2)
Yeast	Class CYT	50.82(4)	53.86(2)	50.75(5)	52.38(3)	**55.56(1)**
	Class NUC	51.79(4)	54.82(3)	60.87(2)	50.77(5)	**83.56(1)**
	Class MIT	50.88(5)	54.71(3)	54.84(2)	52.46(4)	**55.26(1)**
Letter recognition	Class A	51.79(3)	50.66(4)	48.73(5)	52.12(2)	**63.81(1)**
	Class E	52.52(4)	52.65(3)	55.82(2)	51.48(5)	**63.05(1)**
	Class I	53.85(2)	50.85(5)	53.31(4)	53.36(3)	**69.12(1)**
	Class O	43.31(5)	51.45(2)	48.75(3)	48.02(4)	**62.48(1)**
Nursery	Class N	49.64(4)	54.55(3)	56.38(2)	42.58(5)	**74.32(1)**
	Class P	54.04(2)	51.75(3)	50.49(4)	41.23(5)	**72.88(1)**
Chess	Class 13	50.94(3)	50.85(4)	55.21(2)	45.95(5)	**71.99(1)**
	Class 14	50.19(3)	52.87(2)	48.92(4)	41.24(5)	**52.96(1)**
	Class 15	50.47(4)	52.63(3)	50.38(5)	60.88(2)	**64.99(1)**
Average rank		3.10(4)	2.76(2)	3.08(3)	4.08(5)	**1.98(1)**
P value		0	8E−05	0	0	
Wilcoxon test result		1	1	1	1	

The best results are shown in bold. The rankings are included in parentheses

Table 5 The percentage of recall and average rank

Data sets name	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
Breast cancer Wisconsin	Class 2	31.91(2)	27.65(4)	27.66(3)	23.40(5)	**51.06(1)**
	Class 4	41.03(2)	29.07(4)	28.21(5)	30.77(3)	**66.67(1)**
Balance scale	Class 1	57.81(3)	57.61(4)	57.89(2)	47.37(5)	**89.47(1)**
	Class 2	55.56(2)	45.33(4)	48.15(3)	29.63(5)	**70.37(1)**
	Class 3	73.05(2)	53.55(3)	52.73(4)	47.37(5)	**73.68(1)**
Glass identification	Class 1	54.16(4)	56.04(2)	54.82(3)	53.63(5)	**66.67(1)**
	Class 2	45.45(4)	54.38(2)	45.47(3)	36.36(5)	**81.82(1)**
	Class 3	30.77(4)	38.31(2)	37.96(3)	30.71(5)	**38.46(1)**
	Class 5	38.38(3)	39.09(2)	30.77(5)	38.26(4)	**53.85(1)**
	Class 6	45.45(5)	53.45(4)	53.91(2)	53.75(3)	**72.73(1)**
	Class 7	41.67(4)	49.55(2)	41.69(3)	41.27(5)	**50.67(1)**
Ionosphere	Class g	62.35(2)	56.42(5)	61.34(4)	62.05(3)	**81.25(1)**
	Class b	66.46(5)	75.74(2)	**91.67(1)**	74.04(3)	66.67(4)
Blood transfusion service center	Class 0	51.22(2)	39.11(4)	36.59(5)	41.46(3)	**75.61(1)**
	Class 1	37.78(2)	33.84(4)	28.89(5)	35.56(3)	**82.22(1)**
Iris	Class Se	60.01(5)	60.12(4)	79.95(2)	**80.04(1)**	60.13(3)
	Class Ve	57.14(4)	57.76(3)	71.23(2)	**85.71(1)**	57.14(4)
	Class Vi	61.09(3)	59.39(5)	80.24(2)	60.54(4)	**80.42(1)**
Seeds	Class 1	65.62(4)	66.61(3)	64.89(5)	66.67(2)	**66.78(1)**
	Class 2	57.14(5)	72.15(3)	84.53(2)	71.43(4)	**85.71(1)**
	Class 3	62.50(4)	74.33(2)	61.59(5)	62.51(3)	**75.36(1)**
Liver	Class 1	**68.75(1)**	56.99(3)	62.45(2)	56.25(4)	51.12(5)
	Class 2	66.67(5)	74.26(2)	**79.05(1)**	73.21(4)	73.33(3)
SPECTF heart	Class 0	53.63(4)	54.09(2)	53.85(3)	46.15(5)	**69.23(1)**
	Class 1	62.37(5)	**87.86(1)**	87.50(2)	87.21(3)	75.15(4)
Diabetes	Class 1	40.51(3)	38.25(4)	40.54(2)	32.43(5)	**83.78(1)**
	Class 2	57.28(3)	58.13(2)	56.78(4)	39.39(5)	**87.88(1)**
Segment	Class 1	63.22(2)	59.16(3)	58.62(4)	51.72(5)	**79.31(1)**
	Class 2	67.03(2)	61.24(3)	59.34(4)	51.65(5)	**86.81(1)**
	Class 3	59.60(2)	49.33(3)	45.45(4)	37.37(5)	**87.88(1)**
	Class 4	50.51(2)	47.77(3)	46.46(4)	41.41(5)	**87.88(1)**
Vehicle	Class 1	74.36(2)	62.02(4)	64.10(3)	53.85(5)	**97.44(1)**
	Class 2	**90.63(1)**	78.33(4)	81.25(3)	68.75(5)	84.38(2)
	Class 3	83.17(2)	74.33(4)	82.95(3)	67.74(5)	**87.10(1)**
	Class 4	71.79(2)	54.42(5)	56.41(3)	56.32(4)	**79.49(1)**
Musk	Class 0	73.68(4)	73.87(3)	**84.25(1)**	57.89(5)	84.21(2)
	Class 1	**75.19(1)**	65.64(5)	74.57(3)	75.17(2)	70.65(4)
Yeast	Class CYT	52.54(5)	57.24(3)	57.63(2)	55.93(4)	**76.27(1)**
	Class NUC	46.03(5)	56.03(3)	66.67(2)	52.38(4)	**96.83(1)**
	Class MIT	54.72(5)	61.12(3)	64.15(2)	60.31(4)	**79.25(1)**
Letter recognition	Class A	47.08(5)	56.86(3)	54.04(4)	58.54(2)	**77.08(1)**
	Class E	73.85(5)	91.79(2)	**97.37(1)**	89.29(3)	87.34(4)
	Class I	76.02(2)	59.77(4)	60.37(3)	59.38(5)	**91.26(1)**
	Class O	77.38(5)	89.03(4)	90.23(2)	89.13(3)	**92.03(1)**
Nursery	Class N	48.95(3)	45.35(4)	50.21(2)	30.58(5)	**76.30(1)**
	Class P	51.25(2)	43.23(3)	43.18(4)	35.38(5)	**84.96(1)**
Chess	Class 13	63.21(4)	64.17(3)	76.86(2)	56.82(5)	**82.56(1)**
	Class 14	**79.09(1)**	70.72(4)	70.41(5)	71.99(3)	74.16(2)
	Class 15	**70.52(1)**	61.17(4)	61.38(3)	69.96(2)	55.41(5)
Average rank		3.16(3)	3.24(4)	3.00(2)	3.96(5)	**1.61(1)**
P-value		0	0	0	0	
Wilcoxon test result		1	1	1	1	

The best results are shown in bold. The rankings are included in parentheses

Table 6 The percentage of accuracy and average rank

Data sets name	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
Breast cancer Wisconsin	Class 2	32.35(2)	30.88(3)	30.68(4)	29.41(5)	**61.76(1)**
	Class 4	44.12(3)	43.80(4)	42.65(5)	45.59(2)	**60.29(1)**
Balance scale	Class 1	64.52(4)	**72.58(1)**	69.35(2)	67.74(3)	62.79(5)
	Class 2	57.26(4)	58.3(2)	**63.72(1)**	51.61(5)	58.06(3)
	Class 3	68.91(3)	**71.73(1)**	69.35(2)	66.13(4)	63.27(5)
Glass identification	Class 1	61.29(3)	60.69(4)	66.67(1)	52.38(5)	61.90(2)
	Class 2	47.32(4)	52.67(2)	51.31(3)	42.86(5)	**74.36(1)**
	Class 3	38.10(4)	43.40(2)	41.63(3)	33.63(5)	**52.38(1)**
	Class 5	42.26(2)	37.93(4)	33.33(5)	38.11(3)	**52.38(1)**
	Class 6	47.32(4)	46.66(5)	52.38(2)	47.61(3)	**66.67(1)**
	Class 7	45.86(3)	47.04(2)	42.86(4)	42.16(5)	**66.67(1)**
Ionosphere	Class g	57.14(3)	56.84(4)	59.33(2)	54.29(5)	**60.17(1)**
	Class b	**68.57(1)**	62.50(4)	67.57(2)	65.71(3)	57.14(5)
Blood transfusion service center	Class 0	52.13(2)	46.67(4)	45.33(5)	48.37(3)	**68.64(1)**
	Class 1	43.74(4)	44.52(3)	41.33(5)	45.33(2)	**65.36(1)**
Iris	Class Se	66.47(4)	66.67(3)	73.23(2)	**73.33(1)**	60.37(5)
	Class Ve	53.38(3)	53.34(4)	53.33(5)	60.25(2)	**61.57(1)**
	Class Vi	66.63(2)	66.19(3)	66.17(4)	**66.67(1)**	60.75(5)
Seeds	Class 1	71.53(2)	71.23(3)	71.03(4)	**76.19(1)**	66.67(5)
	Class 2	66.48(5)	68.65(3)	68.49(4)	71.14(2)	**71.43(1)**
	Class 3	**66.37(1)**	66.01(2)	61.41(3)	52.38(5)	57.14(4)
Liver	Class 1	57.74(2)	48.57(5)	57.14(3)	54.29(4)	**77.14(1)**
	Class 2	59.73(3)	57.53(5)	60.57(2)	57.64(4)	**65.71(1)**
SPECTF heart	Class 0	54.58(2)	54.36(3)	**55.16(1)**	51.85(4)	51.85(4)
	Class 1	73.57(2)	**74.01(1)**	70.11(4)	70.19(3)	66.67(5)
Diabetes	Class 1	51.95(3)	49.35(4)	53.25(2)	35.06(5)	**64.94(1)**
	Class 2	61.04(2)	**63.04(1)**	58.44(4)	46.75(5)	61.04(2)
Segment	Class 1	60.61(4)	64.50(2)	**64.94(1)**	55.84(5)	62.77(3)
	Class 2	64.94(2)	63.98(3)	62.34(4)	56.28(5)	**65.8(1)**
	Class 3	60.61(2)	60.18(3)	57.14(4)	52.38(5)	**69.26(1)**
	Class 4	55.84(4)	60.13(2)	57.58(3)	45.89(5)	**70.97(1)**
Vehicle	Class 1	58.25(2)	57.86(4)	**62.11(1)**	57.89(3)	55.79(5)
	Class 2	69.18(2)	67.53(3)	**69.47(1)**	60.06(5)	67.37(4)
	Class 3	67.37(4)	67.83(3)	71.58(2)	63.16(5)	**75.79(1)**
	Class 4	61.05(3)	60.94(4)	62.11(2)	58.95(5)	**73.68(1)**
Musk	Class 0	63.18(3)	62.50(4)	**72.92(1)**	54.17(5)	68.75(2)
	Class 1	**65.87(1)**	56.43(5)	60.59(3)	60.12(4)	64.58(2)
Yeast	Class CYT	60.81(4)	62.84(2)	60.81(4)	62.16(3)	**66.22(1)**
	Class NUC	58.78(4)	61.59(3)	67.57(2)	58.11(5)	**90.54(1)**
	Class MIT	64.86(5)	68.32(2)	68.24(3)	65.27(4)	**69.59(1)**
Letter recognition	Class A	56.95(3)	55.85(4)	54.25(5)	57.60(2)	**70.35(1)**
	Class E	66.60(4)	67.07(3)	71.50(2)	65.90(5)	**77.05(1)**
	Class I	68.40(2)	65.66(5)	67.20(3)	66.23(4)	**82.45(1)**
	Class O	75.90(5)	81.07(2)	79.65(3)	79.10(4)	**87.70(1)**
Nursery	Class N	44.60(4)	48.46(3)	51.23(2)	39.12(5)	**72.45(1)**
	Class P	48.84(2)	46.50(3)	45.06(4)	36.27(5)	**74.15(1)**
Chess	Class 13	59.69(3)	59.41(4)	64.72(2)	54.60(5)	**79.54(1)**
	Class 14	82.04(3)	83.32(2)	81.36(4)	76.41(5)	**83.43(1)**
	Class 15	81.15(4)	82.57(3)	81.08(5)	85.67(2)	**85.78(1)**
Average rank		3.00(3)	3.08(4)	2.96(2)	3.90(5)	**2.00(1)**

Table 6 (continued)

Data sets name	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
P-value		0	0	0.00044	0	
Wilcoxon test result		1	1	1	1	

The best results are shown in bold. The rankings are included in parentheses

used for testing while 90% were used for training. In other words, after a new training object arrives and the model is updated, its performance is tested on 10% data left as a test set.

Since the average rank provides a reasonable comparison of the methods, the performance of each method is ranked. The ranks are reported in enclosed parenthesis in Tables 4, 5, 6 and 7. Moreover, one of the popular nonparametric statistical tests called Wilcoxon signed-rank [112] is employed to show the meaningfulness of the experiments. Wilcoxon signed-rank promises rejection of meaningfulness of experiments as the null hypothesis which supposes that there is no noticeable diversity between the proposed method and the others. The proposed method is accepted as a distinguished method if the null hypothesis becomes refused at a 5% significance level. In this test, the significance level of 0.05 is used. The results of the Wilcoxon signed-rank test are reported in two last rows of Tables 4, 5, 6 and 7. The hypothesis test result = 1 indicates a rejection of the null hypothesis at the 0.05 significance level, and the zero value indicates a failure to reject the null hypothesis.

Comparisons of ICCHC with the other methods regarding precision are presented in Table 4. As shown below, ICCHC has an average rank 1. The proposed method for Breast cancer (Wisconsin), Blood transfusion service center, Liver, Yeast, Letter recognition, Nursery, and Chess data sets obtained the best results in all classes. Unfortunately, for Ionosphere (class b), Iris (class Se), Seeds (class 1), SPECT heart (class 5), and Vehicle (class 1), ICCHC did not perform well. In these classes, the boundary of the classifier extends, and more target data will be accepted. However, there will be more non-targets along with these target data within this boundary. Increasing the quantities of target data leads to increasing the amounts of non-targets. Therefore, ICCHC does not perform well in the classes, as mentioned earlier in terms of precision. The remarkable point is that ICCHC achieves good results for data sets with a large number of objects, e.g., Letter recognition, Nursery, and Chess.

The two last rows of Table 4 illustrate the results of the Wilcoxon test comparing the precision criterion of ICCHC against the other competing methods. They demonstrate the superiority of ICCHC in contrast to the others in this case.

The recall is another metric used to judge the performance of the ICCHC. It measures the portion of targets that are correctly identified. Comparison results are reported in Table 5. As shown below, ICCHC achieves the best average rank. The results indicate that in data sets like Breast cancer Wisconsin, Balance scale, Glass identification, Blood transfusion service center, Seeds, Diabetes, Segment, Yeast, and Nursery the proposed method exhibits the best results for all classes. ICCHC gains rank 5 in Liver (class 1) and Chess (class 15). For others, good results have been gained for some classes.

The two last rows of Table 5 list the results of the Wilcoxon test comparing the recall of ICCHC against the other competing methods. They demonstrate there is a significant difference between ICCHC and the other ones.

The third metric for investigation is accuracy, although precision and recall are more appropriate criteria than accuracy for one-class classification. As shown in Table 6, ICCHC has an average rank 1. In Balance scale (classes 1, 3), Ionosphere (class b), Iris (classes Se, Vi), Seeds (class 1), SPECTF heart (class 1), and Vehicle (class 1) ICCHC did not perform well. It means that the number of accepted target data is low because in the space, outside the boundary of the proposed method, there are some target data that this description does not consider them. Therefore, the accuracy of these classes is lower in comparison with the other methods.

The two last rows of Table 6 illustrate the results of the Wilcoxon test comparing the accuracy of the proposed method against the other competing methods. The results demonstrate the superiority of the proposed method in terms of accuracy.

As shown in Table 7, the computational time has also been reported to assess the dynamics of the proposed method. As the average rank shows, A-WOCSVM can significantly reduce the computational time. Both ICCHC and VS-OSVM are ranked 2. S-A-SVDD and C-SVDD are ranked three to four, respectively. Note that, although the A-WOCSVM can be considered as a quick method, it is deficient in producing accurate classifications. Moreover, because of the high precision of ICCHC, this method is relatively time-consuming.

As the two last rows of Table 7 show, there is no significant difference in computational time between ICCHC and C-SVDD, S-A-SVDD, and VS-OSVM. However, ICCHC is significantly different from A-WOCSVM.

To conclude, ICCHC shows higher performance in most data sets regarding the accuracy, precision, and recall. It

Table 7 Average computational time in seconds

Data sets	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
Breast cancer Wisconsin	Class 2	0.68(5)	0.51(2)	0.62(4)	**0.41(1)**	0.58(3)
	Class 4	0.59(4)	0.48(3)	0.45(2)	**0.37(1)**	0.65(5)
Balance scale	Class 1	0.34(4)	0.30(2)	**0.29(1)**	0.33(3)	0.52(5)
	Class 2	0.31(2)	0.39(4)	0.34(3)	**0.21(1)**	0.42(5)
	Class 3	0.39(2)	0.40(3)	**0.35(1)**	0.41(4)	0.42(5)
Glass identification	Class 1	0.67(5)	0.38(3)	0.53(4)	**0.21(1)**	0.36(2)
	Class 2	0.62(5)	**0.31(1)**	0.49(4)	0.32(2)	0.38(3)
	Class 3	0.69(5)	0.44(3)	0.42(2)	0.54(4)	**0.41(1)**
	Class 5	0.72(5)	0.62(3)	0.59(2)	0.63(4)	**0.42(1)**
	Class 6	0.63(5)	0.58(4)	0.46(3)	**0.35(1)**	0.45(2)
	Class 7	0.58(5)	0.53(4)	0.41(2)	0.43(3)	**0.40(1)**
Ionosphere	Class g	0.56(2)	0.75(5)	0.58(3)	**0.55(1)**	0.68(4)
	Class b	0.49(2)	0.68(4)	0.55(3)	**0.45(1)**	0.70(5)
Blood transfusion service center	Class 0	0.52(3)	0.66(5)	0.40(2)	**0.37(1)**	0.55(4)
	Class 1	0.55(4)	0.56(5)	0.34(2)	**0.24(1)**	0.41(3)
Iris	Class Se	0.41(3)	0.50(5)	**0.28(1)**	0.31(2)	0.45(4)
	Class Ve	0.47(4)	0.48(5)	0.32(2)	**0.30(1)**	0.43(3)
	Class Vi	0.51(5)	**0.40(1)**	0.46(4)	0.42(2)	0.44(3)
Seeds	Class 1	0.62(5)	0.50(4)	0.49(3)	0.39(2)	**0.38(1)**
	Class 2	0.66(5)	0.52(4)	0.45(3)	**0.25(1)**	0.39(2)
	Class 3	0.69(5)	0.50(4)	0.38(3)	**0.32(1)**	0.36(2)
Liver	Class 1	0.51(3)	0.78(5)	0.57(4)	0.44(2)	**0.43(1)**
	Class 2	0.53(4)	0.73(5)	0.52(3)	**0.43(1)**	0.44(2)
SPECTF heart	Class 0	0.75(4)	0.54(3)	**0.44(1)**	0.48(2)	0.76(5)
	Class 1	0.72(4)	0.58(3)	**0.46(1)**	0.48(2)	0.85(5)
Diabetes	Class 1	0.62(4)	0.55(3)	0.63(5)	**0.34(1)**	0.54(2)
	Class 2	0.57(3)	0.60(5)	0.58(4)	**0.39(1)**	0.48(2)
Segment	Class 1	0.45(2)	0.55(3)	0.67(4)	**0.29(1)**	0.76(5)
	Class 2	0.45(2)	0.54(3)	0.63(4)	**0.27(1)**	0.75(5)
	Class 3	0.41(2)	0.58(3)	0.62(4)	**0.23(1)**	0.73(5)
	Class 4	0.50(2)	0.54(3)	0.69(5)	**0.21(1)**	0.65(4)
Vehicle	Class 1	0.55(4)	0.45(2)	0.79(5)	**0.39(1)**	0.51(3)
	Class 2	0.61(4)	0.46(3)	0.77(5)	**0.43(1)**	0.44(2)
	Class 3	0.69(4)	0.42(3)	0.82(5)	**0.33(1)**	0.41(2)
	Class 4	0.71(5)	0.61(4)	0.52(3)	**0.29(1)**	0.44(2)
Musk	Class 0	0.78(5)	0.57(2)	0.59(3)	**0.22(1)**	0.75(4)
	Class 1	0.40(2)	0.74(4)	0.67(3)	**0.39(1)**	0.82(5)
Yeast	Class CYT	0.4(2)	0.68(5)	0.66(4)	**0.39(1)**	0.45(3)
	Class NUC	0.43(2)	0.64(4)	0.70(5)	**0.41(1)**	0.47(3)
	Class MIT	2.50(3)	3.02(5)	2.65(4)	2.01(2)	**0.43(1)**
Letter recognition	Class A	2.70(4)	2.63(3)	2.55(2)	**1.54(1)**	3.24(5)
	Class E	**1.60(1)**	2.62(4)	2.73(5)	1.63(2)	2.54(3)
	Class I	2.62(3)	2.52(2)	2.66(4)	1.49(1)	2.96(5)
	Class O	1.77(4)	1.66(3)	**1.49(1)**	1.53(2)	2.66(5)
Nursery	Class N	1.72(5)	1.63(4)	1.56(2)	1.59(3)	**1.53(1)**
	Class P	5.71(5)	5.65(4)	4.59(3)	3.49(2)	**1.51(1)**
Chess	Class 13	4.69(5)	3.63(4)	3.58(3)	**2.39(1)**	3.42(2)
	Class 14	6.66(5)	4.43(2)	5.49(4)	4.45(3)	**3.68(1)**
	Class 15	0.68(4)	0.51(2)	0.62(3)	**0.41(1)**	5.71(5)
Average rank		3.71(4)	3.47(3)	3.12(2)	**1.57(1)**	3.12(2)

Table 7 (continued)

Data sets	Target class	C-SVDD	S-A-SVDD	VS-OSVM	A-WOCSVM	ICCHC
P-value		0.42372	0.56868	0.88076	0	
Wilcoxon test result		0	0	1	1	

The best results are shown in bold. The rankings are included in parentheses

Table 8 Detail of data sets used in noise analysis

Data sets name	#of attributes	#of objects	Target class	#of target	#of others
Breast cancer (Wisconsin)	10	683	Class 2	444	239
Balance scale	4	625	Class 1	49	576
Glass identification	10	214	Class 1	70	144
Ionosphere	33	351	Class g	225	126
Blood transfusion service center	5	748	Class 0	570	178
Iris	4	150	Class Se	50	100
Seeds	7	210	Class 1	70	140
Liver	6	345	Class 1	145	200
SPECTF heart	44	267	Class 0	212	55
Diabetes	8	768	Class 1	500	268
Segment	19	2310	Class 1	330	1980
Vehicle	18	946	Class 1	240	706
Musk	168	476	Class 0	207	269
Yeast	8	1484	Class CYT	463	1021
Letter recognition	16	20,000	Class A	789	19,211
Nursery	8	12,960	Class N	4320	8640
Chess	6	28,056	Class 13	4194	23,862

should be noticed that, although A-WOCSVM can be considered as a quick method, it is deficient in detecting target class.

5.4 Noise analysis

To present a deeper discussion and also prove that ICCHC gains better results, noise analysis was performed on one class of each data set. Used data sets in this experiment don't contain noise themselves. Therefore, both class noise and attribute noise were added manually. The noise analysis has been executed on the benchmark data sets shown in Table 8. To evaluate the impact of noise regarding the accuracy, ICCHC and all the other compared methods were performed on these data sets.

The results of inserting noise on aforesaid data set sare demonstrated in Fig. 11 for each method. The noise level was raised at an interval of 0 to 30 percent. The horizontal axis (x-axis) indicates the noise level, while the vertical axis (y-axis) represents the classification accuracy percentage.

As it is seen in all the diagrams of Fig. 11, when noise is imposed on the data sets, ICCHC shows the least changes in most cases. ICCHC performs well in high-dimensional data sets, e.g., for Segment data set, which has 19 dimensions (attributes), ICCHC has a 23.86% accuracy reduction. Meanwhile, VS-OSVM shows the greatest reduction (40.80%). Also, in 33-dimensional Ionosphere and 44-dimensional SPECTF heart data sets, ICCHC has 13.41% and 8.15% accuracy reduction, respectively. But in such cases, C-SVDD shows 18.71% and 17.30% diminution, respectively. For 168-dimensional Musk data set, ICCHC attains only 14.52% reduction. But in the aforesaid data set, A-WOCSVM shows a 32.68% accuracy reduction. Furthermore, for low-dimensional data sets, e.g., 4-dimensional Balance scale and 4-dimensional Iris, VS-OSVM has slight changes, its changes are 14.03% and 10.88%, respectively.

Moreover, ICCHC performs well in large-scale data sets, e.g., for Yeast (1484 objects) and Chess (28,056 objects), ICCHC presents 10.74% and 21.38% changes, respectively. But in such cases, S-A-SVDD shows 24.65% and 36.87% reduction, respectively. For Segment (2310 objects), ICCHC and VS-OSVM attain the least and highest reduction, respectively. 23.86% reduction is for the former, and 40.80% reduction is for the latter. For Nursery (12,960 objects) and Letter recognition (20,000 objects), ICCHC gains the least changes, 23.27%, and 15.61%, respectively. Meanwhile, A-WOCSVM demonstrates its ineffectiveness in aforesaid data sets. Its

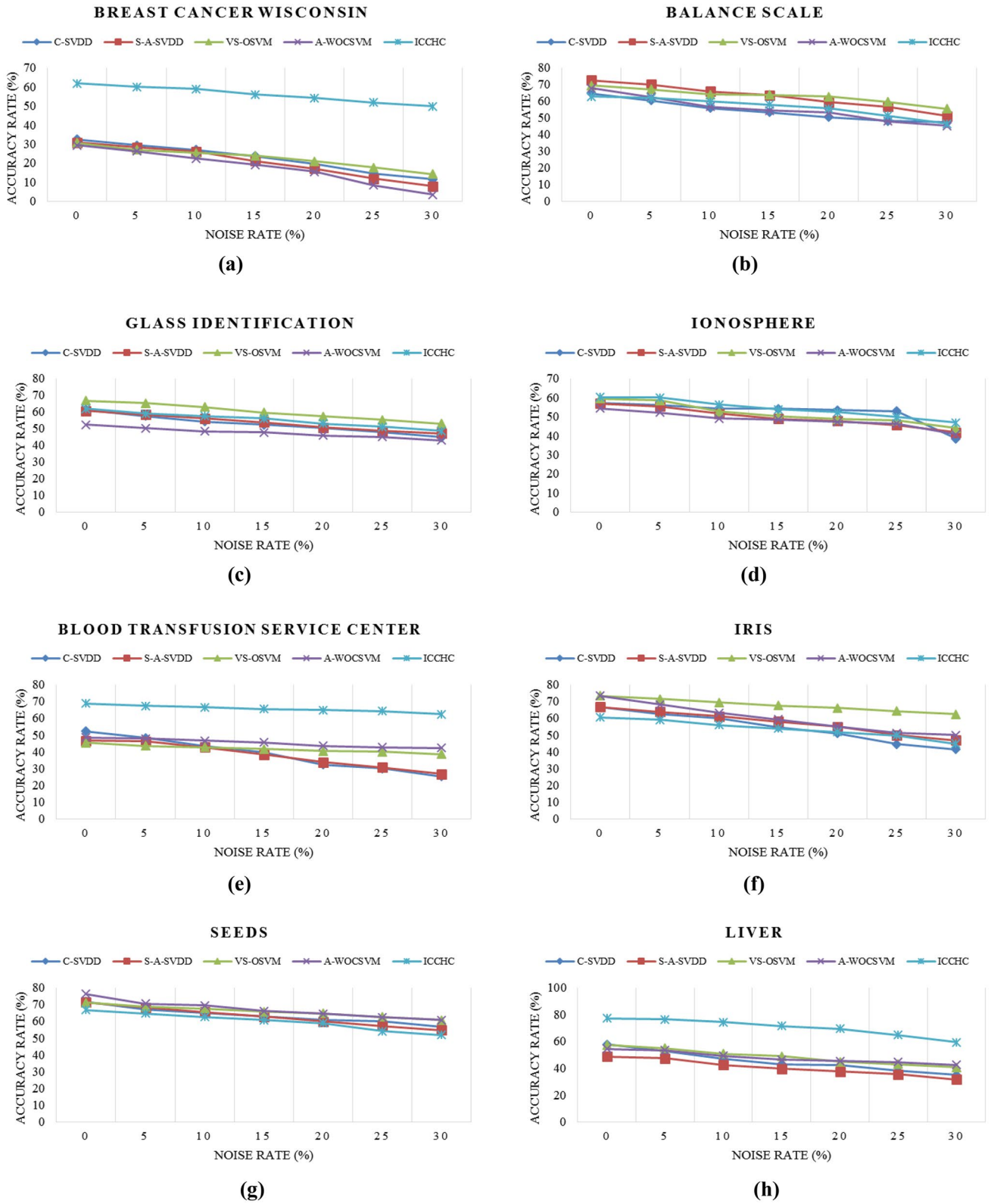

Fig. 11 Experimental results of noise effect on classification accuracy

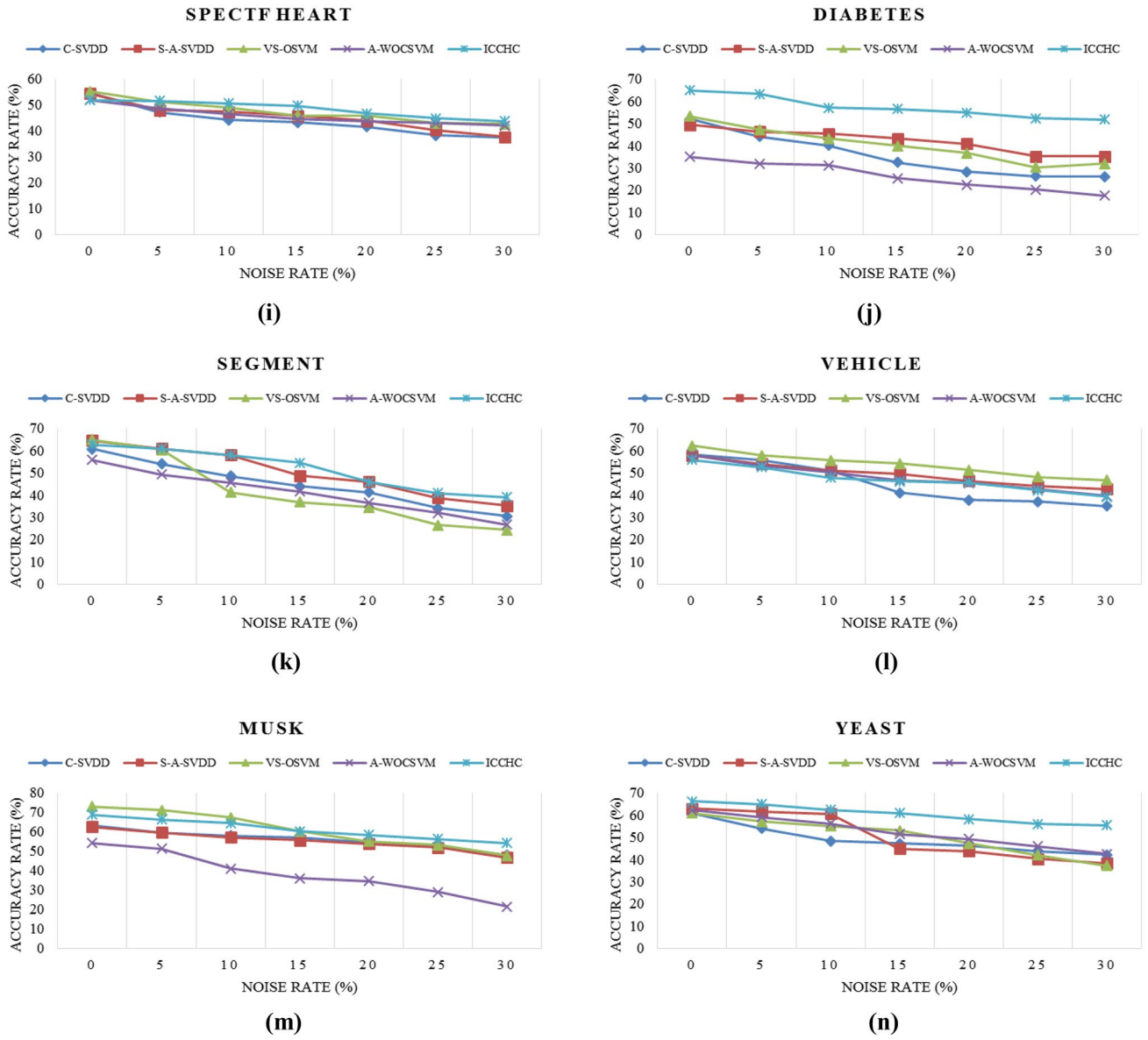

Fig. 11 (continued)

changes are 37.30% and 37.09%, respectively. Furthermore, for small-scale data sets such as Iris (150 objects) and Seeds (210 objects), VS-OSVM performs well. It displays 10.88% and 10.48% changes, respectively.

Briefly, the proposed method is more resistant to the added noise for most data sets. As the noise level increases in each data set, the diminution of the accuracy rate will obviously happen. But the results demonstrate that this diminution is lower for the proposed method in comparison with the other methods. Slight changes indicate the superiority of the proposed method in contrast to the others in the presence of noise.

6 Conclusion and future work

Slight studies on the classification of noisy data streams with concept-drifting motivate us to design a new boundary-based one-class classifier. An incremental approach of the proposed method makes it suitable for online learning tasks on data streams. The proposed method is based on the convex hull to constitute the decision boundary around the target class. It intends to solve the weaknesses of convex hull-based OCC methods by reducing the computational time. Furthermore, it introduces a boundary-based method which is more efficient than the state-of-the-art methods. The primitive boundaries generated by the proposed method are hard. Depending on the data distribution and the

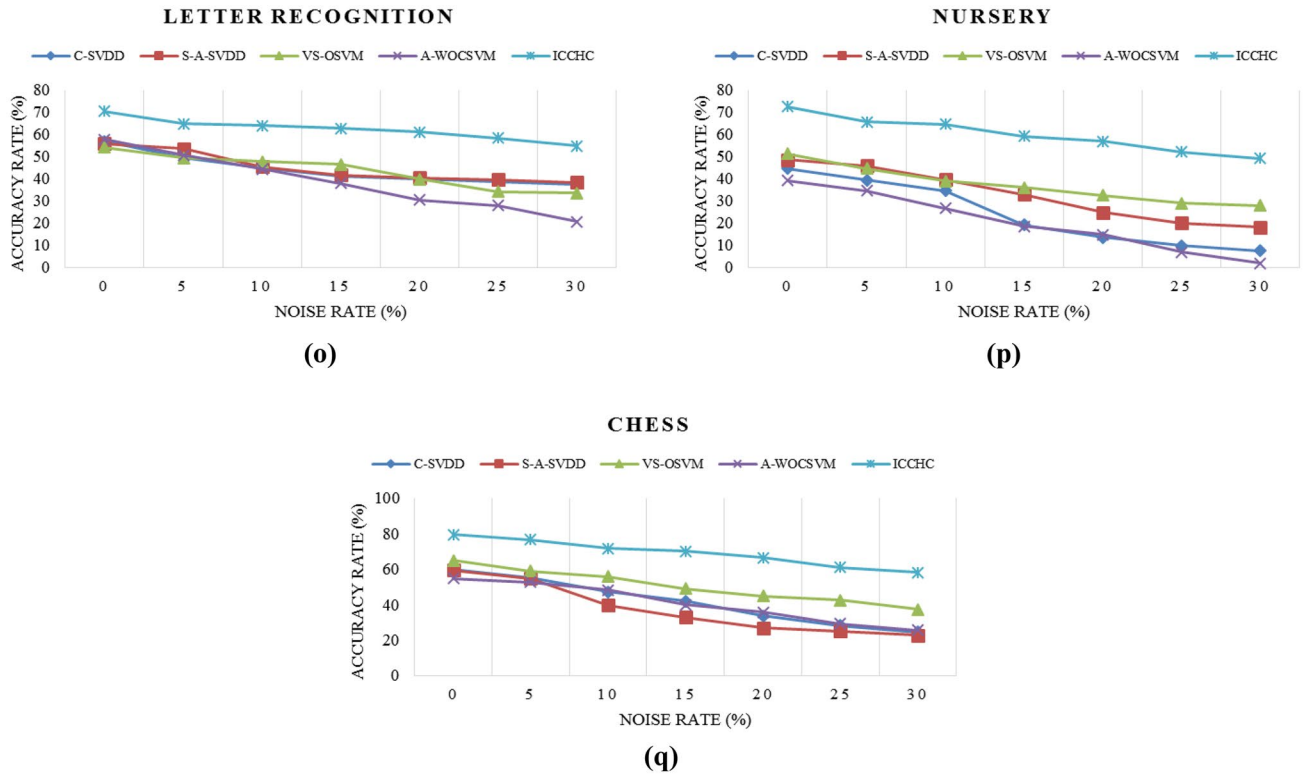

Fig. 11 (continued)

performance of the classifier, it decides whether to perform smoothness or not. In this case, the convex-concave hull can simplify SVDD training. Hence, it does not require complicated computing. Characteristics like using limited data in the construction of convex hulls, deciding on boundary condition (hard/soft), and simplifying SVDD through reducing the training data, causing the proposed method to be mutually beneficial in online situations. Noise analysis was also carried out to evaluate the effectiveness of the proposed method in noise exposure. Experimental results indicate that the proposed method is more resistant to the added noise for most data sets. Its slight changes mean high efficiency in contrast to the others in the presence of noise. According to the results, the proposed method shows better accuracy, precision, recall, and also an acceptable computational time in comparison with the compared methods.

For future work, the authors have decided to focus on one-class classification when there is limited labeled training data. Also, they are interested in exploiting one-class classification to exclude noise from data for multi-class classification.

Compliance with ethical standards

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

Human and animal rights This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

1. Cui L, Shi Y (2014) A method based on one-class SVM for news recommendation. Procedia Comput Sci 31:281–290
2. Moradi M, Hamidzadeh J (2019) Ensemble-based Top-k recommender system considering incomplete data. J AI Data Min 7(3):393–402
3. Bowen RM (2016) Online Novelty Detection System: One-Class Classification of Systemic Operation
4. Manevitz LM, Yousef M (2001) One-class SVMs for document classification. J Mach Learn Res 2(Dec):139–154
5. Chen Y, Hu B, Keogh E, Batista GE (2013) DTW-D: time series semi-supervised learning from a single example. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 383–391
6. Pauwels EJ, Ambekar O (2011) One class classification for anomaly detection: support vector data description revisited. In: Industrial conference on data mining. Springer, pp 25–39
7. Tax DMJ (2001) One-class classification; concept-learning in the absence of counter-examples. Delft University of Technology, Delft

8. Duda RO, Hart PE, Stork DG (2012) Pattern classification. Wiley, Hoboken
9. Bishop CM (1995) Neural networks for pattern recognition. Oxford University Press, Oxford
10. Van Hulle MM (2012) Self-organizing maps. In: Rozenberg G, Bäck T, Kok JN (eds) Handbook of natural computing. Springer, Berlin, Heidelberg, pp 585–622. https://doi.org/10.1007/978-3-540-92910-9_19'
11. Zeng M, Yang Y, Luo S, Cheng J (2016) One-class classification based on the convex hull for bearing fault detection. Mech Syst Signal Process 81:274–293
12. Wang W, Zhang B, Wang D, Jiang Y, Qin S, Xue L (2016) Anomaly detection based on probability density function with Kullback–Leibler divergence. Signal Process 126:12–17
13. Hamidzadeh J, Monsefi R, Yazdi HS (2015) IRAHC: instance reduction algorithm using hyperrectangle clustering. Pattern Recogn 48(5):1878–1889
14. Moghaddam VH, Hamidzadeh J (2016) New Hermite orthogonal polynomial kernel and combined kernels in support vector machine classifier. Pattern Recogn 60:921–935
15. Hamidzadeh J, Moradi M (2018) Improved one-class classification using filled function. Appl Intell 48:1–17
16. Utkin LV, Zhuk YA (2017) An one-class classification support vector machine model by interval-valued training data. Knowl Based Syst 120:43–56
17. Zeng M, Yang Y, Cheng J (2016) A generalized Mitchell–Dem'yanov–Malozemov algorithm for one-class support vector machine. Knowl Based Syst 109:17–24
18. Bhattacharya BK (1982) Application of computational geometry to pattern recognition problems. Ph.d. thesis, School of Computer Science, McGill University
19. Toussaint G (1978) The convex hull as a tool in pattern recognition. In: AFOSR workshop in communication theory and applications
20. Amini S, Homayouni S, Safari A, Darvishsefat AA (2018) Object-based classification of hyperspectral data using Random Forest algorithm. Geo-Spat inf Sci 21(2):127–138
21. Chang Z, Cao J, Zhang Y (2018) A novel image segmentation approach for wood plate surface defect classification through convex optimization. J For Res 29(6):1789–1795
22. Chazelle B (1993) An optimal convex hull algorithm in any fixed dimension. Discrete Comput Geom 10(1):377–409
23. Ruano A, Khosravani HR, Ferreira PM (2015) A randomized approximation convex hull algorithm for high dimensions. IFAC-PapersOnLine 48(10):123–128
24. Chau AL, Li X, Yu W (2013) Convex and concave hulls for classification with support vector machine. Neurocomputing 122:198–209
25. Kodell RL, Zhang C, Siegel ER, Nagarajan R (2012) Selective voting in convex-hull ensembles improves classification accuracy. Artif Intell Med 54(3):171–179
26. Zeng M, Yang Y, Zheng J, Cheng J (2015) Maximum margin classification based on flexible convex hulls. Neurocomputing 149:957–965
27. Khan L, Fan W (2012) Tutorial: data stream mining and its applications. In: International conference on database systems for advanced applications. Springer, pp 328–329
28. Shalev-Shwartz S (2007) Online learning: theory, algorithms, and applications. Thesis submitted for the degree of "Doctor of Philosophy" (Submitted to the Senate of the Hebrew University July 2007, This work was carried out under the supervision of Prof. Yoram Singer)
29. Jiang Y, Shang J, Liu Y (2010) Maximizing customer satisfaction through an online recommendation system: a novel associative classification model. Decis Support Syst 48(3):470–479
30. Sagar B, Singh P, Mallika S (2016) Online transaction fraud detection techniques: a review of data mining approaches. In: 2016 3rd International conference on computing for sustainable global development (INDIACom). IEEE, pp 3756–3761
31. Olszewski D (2012) A probabilistic approach to fraud detection in telecommunications. Knowl Based Syst 26:246–258
32. Olszewski D (2014) Fraud detection using self-organizing map visualizing the user profiles. Knowl Based Syst 70:324–334
33. Cohen Y, Gordon D, Hendler D (2017) Early detection of spamming accounts in large-Scale service provider networks. Knowl Based Syst 142:241–255
34. Wang B, Jones GJ, Pan W (2006) Using online linear classifiers to filter spam emails. Pattern Anal Appl 9(4):339–351
35. Dilmen E, Beyhan S (2017) A novel online LS-SVM approach for regression and classification. IFAC-PapersOnLine 50(1):8642–8647
36. Ding S, Mirza B, Lin Z, Cao J, Lai X, Nguyen TV, Sepulveda J (2018) Kernel based online learning for imbalance multiclass classification. Neurocomputing 277:139–148
37. Suárez-Cetrulo AL, Cervantes A (2017) An online classification algorithm for large scale data streams: iGNGSVM. Neurocomputing 262:67–76
38. Gensler A, Sick B (2018) Performing event detection in time series with SwiftEvent: an algorithm with supervised learning of detection criteria. Pattern Anal Appl 21(2):543–562
39. Soleimani-B H, Lucas C, Araabi BN (2012) Fast evolving neurofuzzy model and its application in online classification and time series prediction. Pattern Anal Appl 15(3):279–288
40. Hamidzadeh J, Moradi M (2020) Enhancing data analysis: uncertainty-resistance method for handling incomplete data. Appl Intell 50(1):74–86
41. Aggarwal CC (2014) A survey of stream classification algorithms. Data classification: algorithms and applications. Springer, New York, USA, pp 245–268
42. Sousa R, Gama J (2018) Multi-label classification from high-speed data streams with adaptive model rules and random rules. Prog Artif Intell 7:1–11
43. Gu X, Angelov PP (2018) Semi-supervised deep rule-based approach for image classification. Appl Soft Comput 68:53–68
44. Verdecia-Cabrera A, Blanco IF, Carvalho AC (2018) An online adaptive classifier ensemble for mining non-stationary data streams. Intell Data Anal 22(4):787–806
45. Ramírez-Gallego S, García S, Herrera F (2018) Online entropy-based discretization for data streaming classification. Future Gener Comput Syst 86:59–70
46. Lobo JL, Del Ser J, Bilbao MN, Perfecto C, Salcedo-Sanz S (2018) DRED: an evolutionary diversity generation method for concept drift adaptation in online learning environments. Appl Soft Comput 68:693–709
47. Feng L-R, Liu C-M, Lai C-C (2018) Probabilistic reverse nearest neighbors on uncertain data streams. In: 2018 7th International symposium on next generation electronics (ISNE). IEEE, pp 1–4
48. Shao X, Zhang M, Meng J (2018) Data stream clustering and outlier detection algorithm based on shared nearest neighbor density. In: 2018 International conference on intelligent transportation, big data and smart city (ICITBS). IEEE, pp 279–282
49. Chatzigeorgakidis G, Karagiorgou S, Athanasiou S, Skiadopoulos S (2018) FML-kNN: scalable machine learning on Big Data using k-nearest neighbor joins. J Big Data 5(1):4
50. Duda P, Jaworski M, Rutkowski L (2018) Convergent time-varying regression models for data streams: tracking concept drift by the recursive parzen-based generalized regression neural networks. Int J Neural Syst 28(02):1750048
51. Prasetyo T, Amar S, Arendra A, Zami MZ (2018) On-line tool wear detection on DCMT070204 carbide tool tip based on noise cutting audio signal using artificial neural network. In:

Journal of physics: conference series, vol 1. IOP Publishing, p 012144
52. van Rijn JN, Holmes G, Pfahringer B, Vanschoren J (2018) The online performance estimation framework: heterogeneous ensemble learning for data streams. Mach Learn 107(1):149–176
53. Brzezinski D, Stefanowski J, Nienkötter A, Jiang X, Last M, Stoliar M, Friedman M, Cornelisse R, Choenni S, Munir M (2018) Ensemble classifiers for imbalanced and evolving data streams. Ser Mach Percept Artif Intell 83(1):44–68
54. Krawczyk B, Cano A (2018) Online ensemble learning with abstaining classifiers for drifting and noisy data streams. Appl Soft Comput 68:677–692
55. Bifet A, Holmes G, Pfahringer B, Kirkby R, Gavaldà R (2009) New ensemble methods for evolving data streams. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 139–148
56. Domingos P, Hulten G (2000) Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 71–80
57. Hulten G, Spencer L, Domingos P (2001) Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 97–106
58. Hashemi S, Yang Y (2009) Flexible decision tree for data stream classification in the presence of concept change, noise and missing values. Data Min Knowl Disc 19(1):95–131
59. Hashemi S, Yang Y, Mirzamomen Z, Kangavari M (2008) Adapted one-versus-all decision trees for data stream classification. IEEE Trans Knowl Data Eng 5:624–637
60. Bifet A, Read J, Holmes G, Pfahringer B (2018) Streaming data mining with massive online analytics (MOA). Ser Mach Percept Artif Intell 83(1):1–25
61. Kang JH, Park CH, Kim SB (2016) Recursive partitioning clustering tree algorithm. Pattern Anal Appl 19(2):355–367
62. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC (2001) Estimating the support of a high-dimensional distribution. Neural Comput 13(7):1443–1471
63. Sabzekar M, Yazdi HS, Naghibzadeh M (2012) Relaxed constraints support vector machine. Expert Syst 29(5):506–525
64. Zhang Y, Chi Z-X (2008) A Fuzzy support vector classifier based on Bayesian optimization. Fuzzy Optim Decis Mak 7(1):75–86
65. Preparata FP, Shamos MI (1985) Introduction. In: Gries D, Schneider F (eds) Computational geometry. Springer, New York, pp 1–35. https://doi.org/10.1007/978-1-4612-1098-6_1
66. Liu Z, Liu J, Pan C, Wang G (2009) A novel geometric approach to binary classification based on scaled convex hulls. IEEE Trans Neural Netw 20(7):1215–1220
67. Casale P, Pujol O, Radeva P (2014) Approximate polytope ensemble for one-class classification. Pattern Recogn 47(2):854–864
68. Bennett KP, Bredensteiner EJ (2000) Duality and geometry in SVM classifiers. In: ICML. pp 57–64
69. Takahashi T, Kudo M (2010) Margin preserved approximate convex hulls for classification. In: 2010 20th International conference on pattern recognition (ICPR). IEEE, pp 4052–4055
70. Pal S, Hattacharya S (2007) Neurocomputing model for computation of an approximate convex hull of a set of points and spheres. IEEE Trans Neural Netw 18(2):600–605
71. Ding S, Nie X, Qiao H, Zhang B (2017) A fast algorithm of convex hull vertices selection for online classification. IEEE Trans Neural Netw Learn Syst 29:792–806
72. Casale P, Pujol O, Radeva P (2011) Approximate convex hulls family for one-class classification. In: International workshop on multiple classifier systems. Springer, pp 106–115
73. Castillo E, Peteiro-Barral D, Berdiñas BG, Fontenla-Romero O (2015) Distributed one-class support vector machine. Int J Neural Syst 25(07):1550029
74. Kemmler M, Rodner E, Wacker E-S, Denzler J (2013) One-class classification with Gaussian processes. Pattern Recogn 46(12):3507–3518
75. Katz A, Thrift P (1993) Hybrid neural network classifiers for automatic target detection. Expert Syst 10(4):243–250
76. Zhang Y, Li X, Orlowska M (2008) One-class classification of text streams with concept drift. In: IEEE international conference on data mining workshops, 2008. ICDMW'08. IEEE, pp 116–125
77. Zhu X, Wu X, Zhang C (2009) Vague one-class learning for data streams. In: Ninth IEEE international conference on data mining, 2009. ICDM'09. IEEE, pp 657–666
78. Zhang D, Cai L, Wang Y, Zhang L (2010) A learning algorithm for one-class data stream classification based on ensemble classifier. In: 2010 International conference on computer application and system modeling (ICCASM). IEEE, pp V2-596–V592-600
79. Li X, Liu B (2003) Learning to classify texts using positive and unlabeled data. In: IJCAI, vol 2003. pp 587–592
80. Liu B, Lee WS, Yu PS, Li X (2002) Partially supervised classification of text documents. In: ICML. pp 387–394
81. Zhu X, Ding W, Philip SY, Zhang C (2011) One-class learning and concept summarization for data streams. Knowl Inf Syst 28(3):523–553
82. Liu B, Xiao Y, Philip SY, Cao L, Zhang Y, Hao Z (2014) Uncertain one-class learning and concept summarization learning on uncertain data streams. IEEE Trans Knowl Data Eng 26(2):468–484
83. Gao K (2015) Online one-class SVMs with active-set optimization for data streams. In: 2015 IEEE 14th International conference on machine learning and applications (ICMLA). IEEE, pp 116–121
84. Saunier N, Midenet S (2013) Creating ensemble classifiers through order and incremental data selection in a stream. Pattern Anal Appl 16(3):333–347
85. Dokur Z (2009) Respiratory sound classification by using an incremental supervised neural network. Pattern Anal Appl 12(4):309
86. Afzal A, Asharaf S (2017) Deep kernel learning in core vector machines. Pattern Anal Appl 21:1–9
87. Wu T, Liang Y, Varela R, Wu C, Zhao G, Han X (2016) Self-adaptive SVDD integrated with AP clustering for one-class classification. Pattern Recogn Lett 84:232–238
88. Jiang Y, Wang Y, Luo H (2015) Fault diagnosis of analog circuit based on a second map SVDD. Analog Integr Circ Sig Process 85(3):395–404
89. Cauwenberghs G, Poggio T (2001) Incremental and decremental support vector machine learning. In: Burges CJC, Smola AJ (eds) Advances in neural information processing systems, pp 409–415
90. Krawczyk B, Woźniak M (2015) One-class classifiers with incremental learning and forgetting for data streams with concept drift. Soft Comput 19(12):3387–3400
91. Bicego M, Figueiredo MA (2009) Soft clustering using weighted one-class support vector machines. Pattern Recogn 42(1):27–32
92. Das B, Cook DJ, Krishnan NC, Schmitter-Edgecombe M (2016) One-class classification-based real-time activity error detection in smart homes. IEEE J Sel Top Signal Process 10(5):914–923
93. Krawczyk B, Woźniak M (2015) Incremental weighted one-class classifier for mining stationary data streams. J Comput Sci 9:19–25
94. Zhou X, Zhang X, Zhang B (2015) An incremental convex hull algorithm based online support vector regression. In: Control conference (CCC), 2015 34th Chinese. IEEE, pp 8220–8225

95. Wang D, Qiao H, Zhang B, Wang M (2013) Online support vector machine based on convex hull vertices selection. IEEE Trans Neural Netw Learn Syst 24(4):593–609
96. Tax DM, Duin RP (2004) Support vector data description. Mach Learn 54(1):45–66
97. Liu B, Xiao Y, Cao L, Hao Z, Deng F (2013) SVDD-based outlier detection on uncertain data. Knowl Inf Syst 34:1–22
98. Yin G, Zhang Y-T, Li Z-N, Ren G-Q, Fan H-B (2014) Online fault diagnosis method based on incremental support vector data description and extreme learning machine with incremental output structure. Neurocomputing 128:224–231
99. Sadeghi R, Hamidzadeh J (2016) Automatic support vector data description. Soft Comput 22:1–12
100. Dubois D, Prade H (1990) Rough fuzzy sets and fuzzy rough sets. Int J Gen Syst 17(2–3):191–209
101. Zadeh LA (1996) Fuzzy sets. In: Klir GJ, Yuan B (eds) Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A Zadeh. World Scientific, pp 394–432
102. Pawlak Z (1982) Rough sets. Int J Parallel Prog 11(5):341–356
103. Radzikowska AM, Kerre EE (2002) A comparative study of fuzzy rough sets. Fuzzy Sets Syst 126(2):137–155
104. Verbiest N, Cornelis C, Herrera F (2013) FRPS: a fuzzy rough prototype selection method. Pattern Recogn 46(10):2770–2782
105. Sinha D, Laplante P (2004) A rough set-based approach to handling spatial uncertainty in binary images. Eng Appl Artif Intell 17(1):97–110
106. Wang QH, Li JR (2004) A rough set-based fault ranking prototype system for fault diagnosis. Eng Appl Artif Intell 17(8):909–917
107. Bárány I (1982) A generalization of Carathéodory's theorem. Discrete Math 40(2–3):141–152
108. Vapnik V (2013) The nature of statistical learning theory. Springer, Berlin
109. Bartlett PL, Mendelson S (2002) Rademacher and Gaussian complexities: Risk bounds and structural results. J Mach Learn Res 3(Nov):463–482
110. Zhu X, Wu X (2004) Class noise vs attribute noise: a quantitative study. Artif Intell Rev 22(3):177–210
111. Chang C-C, Lin C-J (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol (TIST) 2(3):27
112. Sheskin DJ (2003) Handbook of parametric and nonparametric statistical procedures. CRC Press, Boca Raton