Applied Soft Computing 121 (2022) 108712

Contents lists available at ScienceDirect

Applied Soft Computing

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

# Data classification based on attribute vectorization and evidence fusion

Xiaojian Xu [a,b], Xiaobin Xu [b,*], Pengfei Shi [b], Zifa Ye [b], Yu Bai [c], Shuo Zhang [d], Schahram Dustdar [e], Guodong Wang [f]

[a] China Waterborne Transport Research Institute, Beijing 100088, China
[b] School of Automation, Hangzhou Dianzi University, Hangzhou 310018, Zhejiang, China
[c] Tongde Hospital of Zhejiang Province, Hangzhou 310012, Zhejiang, China
[d] Department of Gastroenterology, The First Affliated Hospital, Zhejiang Chinese Medical University, Hangzhou 310006, Zhejiang, China
[e] Research Division of Distributed Systems, Vienna University of Technology, Vienna, 1040, Austria
[f] Shanghai Institute of Computing Technology, Yuyuan Rd 546, Jingan District, Shanghai, 200040, China

ARTICLE INFO

*Article history:*
Received 28 April 2021
Received in revised form 20 February 2022
Accepted 28 February 2022
Available online 17 March 2022

*Keywords:*
Evidential reasoning
Data classification
Principal component analysis
Attribute vectorization

ABSTRACT

Classifiers based on evidential reasoning (ER) rule can well handle the uncertainty in the mapping relationship between input attributes and output classes. To avoid the number of model parameters increasing with the growing number of input attributes, this paper proposes a classification model based on attribute vectorization and evidential reasoning (AV-ER). Firstly, different input attributes are combined into attribute vectors by using principal component analysis (PCA). Then, all training samples are casted into reference attribute vectors , and the reference evidence matrix (REM) is generated by likelihood function normalization. After that, all pieces of activated evidence are fused through ER theory to generate the final classification decision. In the fusion process, parameters of the initial classification model are optimized by genetic algorithm (GA), and Akaike information criterion (AIC) is used to evaluate the model performance comprehensively considering the model complexity and classification accuracy. Finally, typical UCI benchmark datasets are applied to verify the proposed AV-ER classification model, and the results indicate that the classification performance of the AV-ER model is satisfying while the number of the model parameters decrease obviously as well.



## 1. Introduction

In machine learning and statistics, data classification is one of the most important topics [1]. It aims to determine the class of a new observed sample by training a model based on data samples with the known class. To be specific, suppose there are $J$ input attributes $\{x_1, x_2, \ldots, x_J\}$ and $P$ output classes $\{y_1, y_2, \ldots, y_P\}$ in a classification problem, the essential of a classifier is to build a linear or nonlinear relationship between the input $\{x_1, x_2, \ldots, x_J\}$ and its corresponding output class. Data classification has been widely applied in variety of field, such as medical diagnosis [2], image process [3], fault diagnosis [4], risk analysis [5], cyber security [6] and etc. As a hot topic, many algorithms are proposed to solve classification problems, including artificial neural network [7], factor analysis [8], support vector machine (SVM) [9], naive bayes [10], decision tree [11], and other new technologies, such as deep learning [12], belief rule-based inference methodology [13]. In practical, different classifiers have their own advantages in dealing with different fields of data. However, almost all methods have to face uncertainty problem in classification. From the perspective of uncertain information processing, the inaccurate or wrong classification result is caused by the uncertain attribute boundaries among different classes, thus the class of a sample cannot be determined with its input attributes [14–16].

Dempster–Shafer (DS) evidence theory offers an effective mechanism to deal with uncertainty in classification. Firstly, DS theory defines a framework of discernment (FoD) containing all classes in a problem. Then, a basic belief assignment (BBA) is determined based on the observed values of attributes, which is a belief distribution function on the power set element of the FoD. BBA represents the belief degree that every element and subset of the FoD occurs, and it is named evidence in evidential reasoning (ER) theory. The belief distribution given by BBA is different from the traditional probability distribution. The former distribution can assign the belief to a single proposition of the FoD and any subset composed by the propositions as well, which consequently is the most natural and flexible generalization of the latter distribution. Currently, various methods can be used

* Corresponding authors.
*E-mail address:* xuxiaobin1980@163.com (X. Xu).

https://doi.org/10.1016/j.asoc.2022.108712
1568-4946/

to generate BBA from different kinds of attribute information, such as kernel samples [14], neural network [15], $k$-NN [16] and expert system [13]. Finally, all pieces of evidence offered by attributes are fused by evidence combination rules, and the class of a sample is made according to the fusion result. The aim of evidence fusion is to reduce the uncertainty in classification through fusing multi-source information.

Evidential reasoning (ER) rule is the extension of DS evidence theory, and can also solve the classification problem [17]. Many researchers further use ER rule for classification problems in various areas by using quantity data or qualitative information [17–19]. In traditional ER rule, every attribute is assigned a reliability factor $r$ and an importance weight $w$, and has its own REM. Consequently, when ER rule is used to deal with samples having high dimensional attributes, the model structure becomes more complex with the rising number of input attributes. Meanwhile, the fusion times of evidence, the number of model parameters, and the calculation amount increase as well.

To control the structural complexity and calculation amount of ER classifiers for high dimensional input attributes, this paper proposes a new ER classifier on the basis of attribute vectorization (i.e. AV-ER classifier), which effectively extends the flexibility of evidential reasoning for general classification problems. Firstly, compared with other methods for feature reduction as studied in Ref. [20], PCA is adopted to calculate the importance weight of every input attribute with the training dataset, and the high dimensional attributes are decomposed into multiple attribute vectors according to the training dataset. Meanwhile, the reliability factor of every attribute vector $r$ is determined. Secondly, the method on fine tuning the importance weight $w$ of the AV-ER classifier is described, and the optimized classifier is used to identify the classes of testing data samples. Finally, typical databases in machine learning are used to verify the performance of the proposed classifier. Under the AIC criteria, the AV-ER classifier and the ER classifier are compared in detail from structural complexity and classification accuracy, and the results indicate that the new classifier has better comprehensive performance.

The main contributions of this paper are as follows. Firstly, the paper proposes a method to vectorize the input attributes through calculating the importance of every input attribute to the principal components based on PCA. In the process, any input attribute is neither deleted nor transformed into other components. Secondly, determining referential values and generating REM for every attribute in ER rule method are improved so that it is appropriate for the input attribute vectors. Lastly, the AV-ER model proposed in this paper increases the flexibility of ER rule method, especially when ER rule is applied in high-dimension problems. The complexity of the AV-ER model can be reduced compared with ER rule method, while the AV-ER model still has a good interpretability.

## 2. Theory on ER rule

Suppose $\Theta$={$y_1$, $y_2$,…, $y_P$} is the FoD, where $y$ is the proposition to be studied, which is the category label in a classification problem. Every proposition is mutually exclusive and collective with each other. The power set of $\Theta$ is represented by $P(\Theta)$ or $2^{\Theta}$. In ER rule, a piece of evidence for attribute $x_j$ which is extracted from observed samples can be profiles by Eq. (1).

$$e_j = \{(\theta, p_{\theta,j}) | \forall \theta \subseteq \Theta, \sum\nolimits_{\theta \in \Theta} p_{\theta,j} = 1\} \tag{1}$$

Where $(\theta, p_{\theta,j})$ represents that evidence $e_j$ supports the proposition $\theta$ with the belief degree $p_{\theta,j}$. $\theta$ can be any element of $P(\Theta)$ or any subset of $\Theta$.

ER rule defines the reliability factor $r_j$ and importance weight $w_j$ of evidence $e_j$ (i.e. input attribute $x_j$). Specifically, $r_j$ represents the ability of the information source $x_j$ for $e_j$ to offer the accurate assessment for a specific problem. $w_j$ defines the relative importance of $e_j$ compared with other evidence, which depends on the evidence to be fused, the evidence users, and the application scenarios. From the above definitions, it is known that the reliability factor and importance weight are totally different. $r_j$ is the inherent property of evidence which purely depends on the reliability of information source, while $w_j$ is subjective and depends on the information sources of other evidence in the fusion process.

The evidence modified by $r_j$ and $w_j$ which is also called BBA is defined as follows [21]:

$$m_j = \{(\theta, \tilde{m}_{\theta,j}) | \forall \theta \subseteq \Theta; (P(\Theta), \tilde{m}_{P(\Theta),j})\} \tag{2}$$

Where $\tilde{m}_{\theta,j}$ measures the supporting degree of $e_j$ to $\theta$ considering $r_j$ and $w_j$, and it is defined as Eq. (3).

$$\tilde{m}_{\theta,j} = \begin{cases} 0 & \theta = \varnothing \\ c_{rw,j} m_{\theta,j} & \theta \subseteq \Theta, \theta \neq \varnothing \\ c_{rw,j}(1 - r_j) & \theta = P(\Theta) \end{cases} \tag{3}$$

In Eq. (3), $m_{\theta,j} = w_j p_{\theta,j}$, $c_{rw,j} = 1/(1 + w_j - r_j)$ is a normalization factor, ensuring $\sum_{\theta \in \Theta} \tilde{m}_{\theta,j} + \tilde{m}_{P(\Theta),j} = 1$ given that $\sum_{\theta \in \Theta} p_{\theta,j} = 1$. ER rule defines the residual support degree (1-$r_j$) discounted by reliability factor as the unreliability of evidence, and it is assigned to the power set $P(\Theta)$, indicating that it may support the universal set $\Theta$ or any subset of $\Theta$. This discounting method ensures $e_j$ and $m_j$ have the same probability characteristics, that is to say the relative ratio among the belief degree of every $\theta$ in $e_j$ are the same with that in $m_j$.

For the mutual independent evidence $e_1$ and $e_2$ from two different information sources, the belief degree of $e_1$ and $e_2$ jointly supporting $\theta$ which is represented by $p_{\theta,e(2)}$ can be generated by ER fusion as shown in Eq. (4).

$$e(2) = \{(\theta, p_{\theta,e(2)}) | \forall \theta \subseteq \Theta, \sum\nolimits_{\theta \subseteq \Theta} p_{\theta,e(2)} = 1\} \tag{4a}$$

$$p_{\theta,e(2)} = \begin{cases} 0 & \theta = \varnothing \\ \dfrac{\hat{m}_{\theta,e(2)}}{\sum_{D \subseteq \Theta} \hat{m}_{D,e(2)}} & \theta \subseteq \Theta, \theta \neq \varnothing \end{cases}$$

$$\hat{m}_{\theta,e(2)} = [(1 - r_2) m_{\theta,1} + (1 - r_1) m_{\theta,2}] + \sum_{B \cap C = \theta} m_{B,1} m_{C,2} \quad \forall \theta \subseteq \Theta \tag{4b}$$

From the above description, it can be known that each piece of evidence is modified by $r_j$ and $w_j$ at first in ER rule to generate the belief distribution function considering reliability factor and importance weight of evidence. After that, two independent belief distribution functions are fused by ER rule to achieve the joint supporting degree of the two pieces of independent evidence to a certain proposition in FoD.

For $J$ pieces of evidence {$e_1$, $e_2$, …, $e_J$} from $J$ information sources which are independent with each other, Eq. (4b) can be generalized to combine multiple pieces of evidence, and the fused evidence is denoted as Eq. (5).

$$e(J) = \{(\theta, p_{\theta,e(J)}) | \forall \theta \subseteq \Theta, \sum\nolimits_{\theta \subseteq \Theta} p_{\theta,e(J)} = 1\} \tag{5a}$$

$$p_{\theta,e(J)} = \begin{cases} 0 & \theta = \varnothing \\ \dfrac{\hat{m}_{\theta,e(J)}}{\sum_{A \subseteq \Theta} \hat{m}_{A,e(J)}} & \theta \neq \varnothing \end{cases} \tag{5b}$$

In Eq. (5b), $\hat{m}_{\theta,e(J)}$ is acquired after the recursive fusion by ER rule as illustrated by Eq. (6a), $m_{\theta,e(j-1)}$ and $m_{P(\Theta),e(j-1)}$ ($j$ = 1,2,…, $J$) in Eq. (6a) can be generated by Eq. (6b) and Eq. (6c)

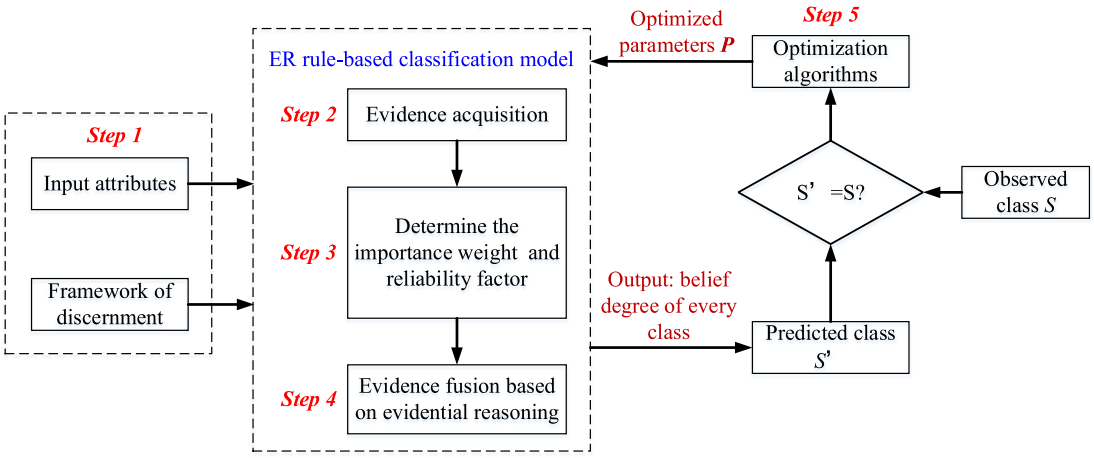


**Fig. 1.** The algorithm process of ER rule-based classification.

respectively.

$$\hat{m}_{\theta,e(j)} = [(1-r_j)m_{\theta,e(j-1)} + m_{P(\Theta),e(j-1)}m_{\theta,j}] + \sum_{B\cap C=\theta} m_{B,e(j-1)}m_{C,j}, \theta \subseteq \Theta \tag{6a}$$

$$m_{\theta,e(j-1)} = [m_1 \oplus \cdots \oplus m_{j-1}](\theta) = \begin{cases} 0 & \theta = \varnothing \\ \dfrac{\hat{m}_{\theta,e(j-1)}}{\sum_{D\subseteq\Theta} \hat{m}_{D,e(j-1)} + \hat{m}_{P(\Theta),e(j-1)}} & \theta \neq \varnothing \end{cases} \tag{6b}$$

$$\hat{m}_{P(\Theta),e(j-1)} = (1 - r_{j-1})m_{P(\Theta),e(j-2)} \tag{6c}$$

In Eq. (6b), $m_j$ represents the $j$th piece of evidence to be fused, and is calculated by Eq. (2). In the iterative calculation, $\sum_{\theta\subseteq\Theta} m_{\theta,e(j)} + m_{P(\Theta),e(j)} = 1, j = 1, 2, \ldots, J$.

When ER rule method is applied for classification, the algorithm process can be divided into five steps as shown in Fig. 1. Firstly, the input attributes and FoD (i.e. the output classes of the ER rule model) should be determined. Secondly, to acquire evidence, we should determine the reference values of every attribute and generate its corresponding REM based on the historical data or expert experience. Specifically, likelihood function normalization is an effective method to generate REMs. Then, the importance weight and reliability factor of evidence should be determined. Different methods have been proposed to determine reliability factors, as Refs. [17,19]. The importance weight of evidence is generally fine-tuned by optimization algorithms, such as genetic algorithm (GA). After that, all pieces of evidence will be fused by evidential reasoning algorithm. Each piece of evidence corresponds to an attribute. Finally, an optimization model is constructed, and the parameters of ER rule model including importance weight of evidence will be trained by optimization algorithms based on historical data. The optimized ER rule model will be used to identify the class of a new sample. Ref. [17] presents the detailed process of ER rule-based classification. To easily understand the large number of symbolic variables contained in this paper, the abbreviations and detailed explanations of these symbolic variables are given in Appendix A.

## 3. Classifier via attribute vectorization and ER rule

Assume that the sample set contains $T$ samples in a classification problem, and each sample has $J$ attributes $x$={$x_1$, $x_2$, …, $x_J$}, pointing to $P$ classes. Then, the FoD is $\Theta$={$y_1$, …, $y_p$,…, $y_P$}, where $y_p$ denotes that the sample belongs to the $p$th class. This paper uses the $J$ attributes as the input of AV-ER classifier, and the output are the $P$ classes.

For the dataset containing $J$ attributes {$x_1$, $x_2$, …, $x_J$}, the evidence $e_j$ corresponding to every attribute $x_j$, the reliability factor of the evidence $r_j$, and the importance weight of the evidence $w_j$ are essential for the utilization of ER rule. If more attributes are involved in the classification problem, the amount of evidence to be fused, the number of parameters $r_j$ and $w_j$ to be optimized, and the fusion time will increase, causing the rising calculation amount. Therefore, the AV-ER classifier can effectively reduce the amount of evidence and the number of the model parameters in the premise that the classification accuracy can be ensured.

Fig. 2 is the flowchart of AV-ER classifier. As illustrated in Fig. 2, the proposed method can be divided into four steps:

(1) The generation of attribute vectors based on PCA. In order to reduce the input feature dimension and ensure the good interpretability of the model. Calculate the contributions of principal components to acquire the reliability factor of every attribute, and rank the reliability factors to transform the $J$ attributes into $K$ attribute vectors. Meanwhile, determine the reliability factor $r$ of every attribute vector.

(2) Generate REM through statistically analyzing samples of attribute vectors. Cast the training samples into reference vectors, and acquire the REM by likelihood function normalization.

(3) Inference process based on ER rule. The attribute vectors of every testing sample will act all reference evidence in REM. All pieces of active evidence are fused by evidential reasoning, and the classification decision is made according to the fusion result.

(4) Optimize the parameters of AV-ER classifier. To improve the accuracy of the model, a genetic algorithm was used to fine-tune the parameters of the initial AV-ER model to achieve the best performance.

After the four steps, the classifier can be evaluated based on AIC. Vary the number of attribute vectors $K$ and evaluate the structural complexity and classification accuracy of the classifier by using AIC. Table B.1 list the pseudocode of the attribute vectorization based on PCA, and the inference process based on ER rule which includes REM generations for attribute vectors, inference process, parameters optimization of AV[HYPHEN]ER classifier, and performance evaluation of the classifier based on AIC can be refered to Table B.2.

### 3.1. Attribute vectorization based on PCA

As a multivariate statistics, PCA can realize the multi-dimensional orthometric linear transformation, which is generally used to extract features. PCA has been widely applied

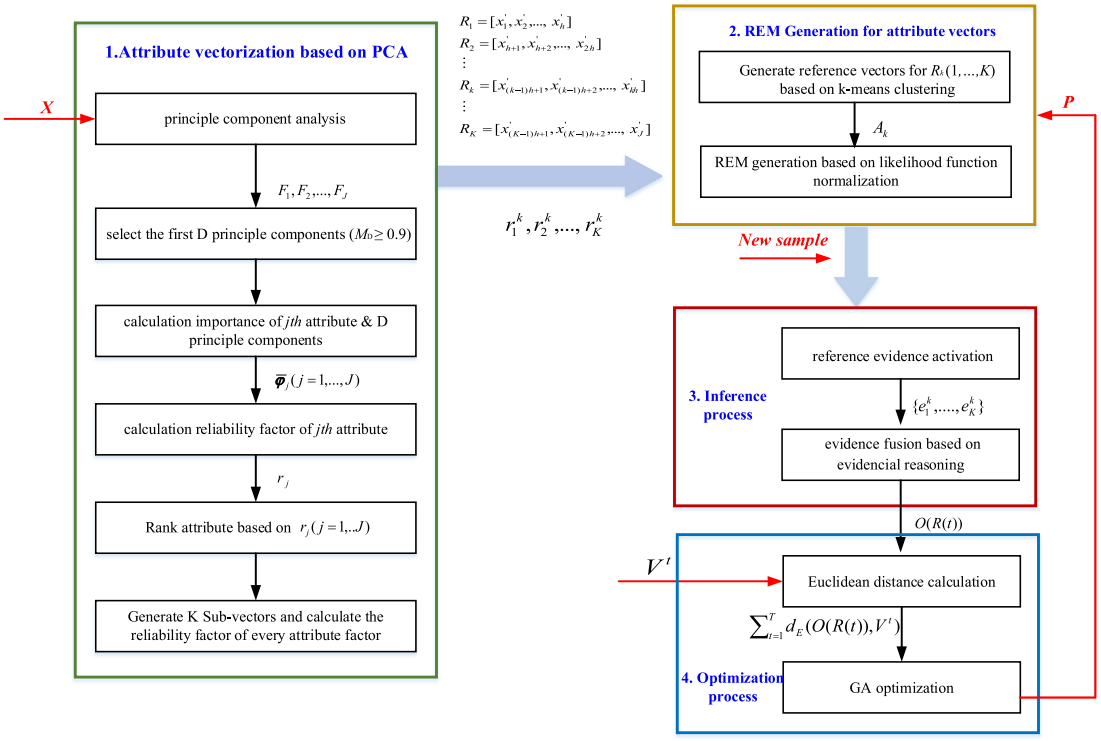


**Fig. 2.** The flowchart of AV-ER classifier.

in pattern recognition, image processing, chemistry, and etc. The variable obtained by PCA (i.e principal component) is a linear combination of the original variables, and the principal components are sorted from largest to smallest by their contributions [22]. With PCA, the important features in high dimensional data can be reserved, while the noise and less important features can be removed. As a result, the data processing speed can be accelerated. In this section, we will fully use the characteristic of PCA that the principal components are ranked by contributions to sort the importance of every original attribute. Based on the sorted result, $K$ attribute vectors with different importance weights are acquired. The detailed processes are as follows:

**Step 3.1.1:Transformation from attribute variables to principal components**

In the dataset $X=\{x_1, \ldots, x_j, \ldots, x_J\}$, $x_j=[x_j(1), \ldots, x_j(t), \ldots, x_j(T)]^{\mathrm{T}}$ is the vector containing $T$ samples for the $j$th attribute variable, and $T$ represents transposition. Use PCA to process the dataset in order to map the $J$ attributes to the principal components space. Consequently, $J$ principal components can be acquired as Eq. (7).

$$\begin{cases} F_1 = w_{11}x_1 + w_{21}x_2 + \cdots + w_{J1}x_J \\ F_2 = w_{12}x_1 + w_{22}x_2 + \cdots + w_{J2}x_J \\ \qquad\qquad \vdots \\ F_i = w_{1i}x_1 + w_{2i}x_2 + \cdots + w_{ji}x_J \\ \qquad\qquad \vdots \\ F_J = w_{1J}x_1 + w_{2J}x_2 + \cdots + w_{JJ}x_J \end{cases} \tag{7}$$

Here $F_i = w_{1i}x_1 + w_{2i}x_2 + \ldots + w_{ji}x_j + \ldots + w_{Ji}x_J$, where $x_j$ is a $T$-dimensional vector, and $w_{ji}$ denotes the weighting coefficient of the $j$th attribute to the $i$th principal component which is also a $T$-dimensional vector. The principal components should meet the following constrains: (1) $F_i$ and $F_j$ ($i \neq j$; $i, j = 1, 2, \ldots, J$) are uncorrelated; (2) the variances of the principal components are in decreasing order, i.e. the variance of $F_1$ is bigger than that of $F_2$, the variance of $F_2$ is bigger than that of $F_3$, and so on. Then, the transformation matrix $W$ composed of weighting coefficients can be given, and with $W$, the formula calculating principal component is $F=[F_1, F_2, \ldots, F_J]= XW$.

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1i} & \cdots & w_{1J} \\ w_{21} & w_{22} & \cdots & w_{2i} & \cdots & w_{2J} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{J1} & w_{J2} & \cdots & w_{Ji} & \cdots & w_{JJ} \end{bmatrix}$$

**Step 3.1.2: Calculate the contribution of principal component and the attribute reliability**

To acquire the principal component of the original attribute variables, these samples in dataset $X$ should be centralized according to Eq. (8).

$$\tilde{x}_j(t) = x_j(t) - \mu_j \tag{8}$$

Where $\mu_j$ is the mean value of $x_j$, and can be calculated as Eq. (9).

$$\mu_j = \frac{1}{T}\sum_{t=1}^{T} x_j(t) \tag{9}$$

The matrix after centralization is represented by

$$\tilde{X} = \begin{bmatrix} \tilde{x}_1(1) & \tilde{x}_2(1) & \cdots & \tilde{x}_j(1) & \cdots & \tilde{x}_J(1) \\ \tilde{x}_1(2) & \tilde{x}_2(2) & \cdots & \tilde{x}_j(2) & \cdots & \tilde{x}_J(2) \\ \vdots & \vdots & & \vdots & & \vdots \\ \tilde{x}_1(t) & \tilde{x}_2(t) & & \tilde{x}_j(t) & & \tilde{x}_j(t) \\ \vdots & \vdots & & \vdots & & \vdots \\ \tilde{x}_1(T) & \tilde{x}_2(T) & \cdots & \tilde{x}_j(T) & \cdots & \tilde{x}_J(T) \end{bmatrix}$$

And the correlation coefficient matrix of $\tilde{X}$ is denoted by Eq. (10).

$$V = \frac{1}{T-1}\tilde{X}^T\tilde{X} \tag{10}$$

Through eigenvalue decomposition for $V$, $J$ eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_J$ and their corresponding eigenvector $\omega_j$(j=1,2,..., $J$) can be obtained, and the contribution of the $j$th principal component can be calculated by Eq. (11).

$$\varphi_j = \frac{\lambda_j}{\sum_{j=1}^{J} \lambda_j} \tag{11}$$

In practical application, the first $D$ ($D$<$J$) principal components are selected, making the cumulative variance contribution over 0.9, i.e. $M_D \geq 0.9$. The larger cumulative contribution indicates that the first $D$ principal components can better cover the information in the original data samples. Eq. (12) is the formula to calculate the cumulative variance contribution.

$$M_D = \frac{\sum_{i=1}^{D} \lambda_i}{\sum_{i=1}^{J} \lambda_i} \tag{12}$$

The eigenvalues corresponding to the first $D$ principal components $\Lambda = diag[\lambda_1, \lambda_2, \ldots, \lambda_D]$ and their eigenvector $W_D = [\omega_1, \omega_2, \ldots, \omega_D]$ are used as the base of the subspace. The extracted $D$ principal components are as Eq. (13).

$$F_D = \tilde{X}W_D \tag{13}$$

Then, the reliability factors of attribute variables can be determined. Since every principal component has its own contribution as Eq. (11), and the weighting coefficient of the $j$th attribute to the $i$th principal component $w_{ji}$ ($i = 1, 2, \ldots, D, j = 1, 2, \ldots, J$) can be acquired by Eq. (7), the importance of the $j$th attribute to all principal components can be calculated by Eq. (14a) by comprehensively considering the coefficient of the principal component $w_{ji}$ and its contribution $\varphi_j$.

$$\overline{\varphi}_j = \sum_{i=1}^{D} \varphi_i w_{ji} \tag{14a}$$

The reliability factor of the $j$th attribute can be determined with the importance $\overline{\varphi}_j$ according to Eq. (14b).

$$r_j = \frac{\overline{\varphi}_j}{\max\limits_{j=1,\ldots,J}(\overline{\varphi}_j)} \tag{14b}$$

**Step 3.1.3: Attribute vectorization based on attribute importance**

Rank the attributes from large to small and renumber them as $x'_1, x'_2, \ldots, x'_J$. After that, the $J$ attributes in the new order are divided into $K$ sub-vectors as follows:

$$R_1 = [x'_1, x'_2, \ldots, x'_h]$$
$$R_2 = [x'_{h+1}, x'_{h+2}, \ldots, x'_{2h}]$$
$$\vdots$$
$$R_k = [x'_{(k-1)h+1}, x'_{(k-1)h+2}, \ldots, x'_{kh}]$$
$$\vdots$$
$$R_K = [x'_{(K-1)h+1}, x'_{(K-1)h+2}, \ldots, x'_J]$$

Where $K$ is the total number of the attribute vectors, and $h$ is the number of attributes that every vector contains. The reliability factor of the attribute vector $R_k$ can be represented by $r_k^R$ and defined by Eq. (15).

$$r_k^R = \frac{1}{h}\sum_{j=(k-1)h+1}^{kh} r_j \tag{15}$$

**Table 1**
Casting result of samples ($R_k$, $y$) on attribute vector $R_k$.

| $y$ | $R_k$ | | | | |
|---|---|---|---|---|---|
| | $A_1^k$ | ... | $A_n^k$ | ... | $A_N^k$ |
| $y_1$ | $a_{1,1}$ | ... | $a_{1,n}$ | ... | $a_{1,N}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $y_p$ | $a_{p,1}$ | ... | $a_{p,n}$ | ... | $a_{p,N}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $y_P$ | $a_{P,1}$ | ... | $a_{P,n}$ | ... | $a_{P,N}$ |

**Table 2**
REM for input attribute vector $R_k$.

| $y$ | $R_k$ | | | | |
|---|---|---|---|---|---|
| | $e_1^k$ | ... | $e_n^k$ | ... | $e_N^k$ |
| | $A_1^k$ | ... | $A_n^k$ | ... | $A_N^k$ |
| $y_1$ | $\mu_{1,1}^k$ | ... | $\mu_{1,n}^k$ | ... | $\mu_{1,N}^k$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $y_p$ | $\mu_{p,1}^k$ | ... | $\mu_{p,n}^k$ | ... | $\mu_{p,N}^k$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $y_P$ | $\mu_{P,1}^k$ | ... | $\mu_{P,n}^k$ | ... | $\mu_{P,N}^k$ |

### 3.2. REM generation for attribute vectors

According to the attribute vectors, rearrange the dataset $X$ used in PCA to generate the training dataset $U$={$R(t)$=[($R_1(t)$, ..., $R_k(t)$, ..., $R_K(t)$), $y(t)$]| $t = 1, 2, \ldots, T$, $y(t) \in \Theta$}. With the likelihood function normalization described in Ref. [23], generate the reference evidence of every attribute vector from the training dataset $U$ as illustrated in the following.

**Step 3.2.1: Determine the reference vector of every attribute vector by *k*-means clustering.**

In this section, $k$-means is used to determine the reference vectors of the $K$ attribute vectors in Section 3.1. Every sample $R_k(t)$ in the training dataset $U$ is divided into the cluster which

**Table 3**
The reliability factors of input attributes.

| $x'_1$ ($x_{14}$) | $x'_2$ ($x_{32}$) | $x'_3$ ($x_8$) | $x'_4$ ($x_{10}$) | $x'_5$ ($x_{12}$) | $x'_6$ ($x_{28}$) | $x'_7$ ($x_{24}$) | $x'_8$ ($x_{18}$) | $x'_9$ ($x_{16}$) | $x'_{10}$ ($x_{22}$) | $x'_{11}$ ($x_6$) |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.9675 | 0.9635 | 0.9628 | 0.9513 | 0.9481 | 0.9434 | 0.9433 | 0.9424 | 0.9412 | 0.9377 |
| $x'_{12}$ ($x_{20}$) | $x'_{13}$ ($x_{30}$) | $x'_{14}$ ($x_{26}$) | $x'_{15}$ ($x_7$) | $x'_{16}$ ($x_9$) | $x'_{17}$ ($x_4$) | $x'_{18}$ ($x_{13}$) | $x'_{19}$ ($x_{11}$) | $x'_{20}$ ($x_{15}$) | $x'_{21}$ ($x_{27}$) | $x'_{22}$ ($x_{21}$) |
| 0.9358 | 0.9350 | 0.9144 | 0.8644 | 0.8550 | 0.8229 | 0.7996 | 0.7976 | 0.7645 | 0.7373 | 0.7314 |
| $x'_{23}$ ($x_2$) | $x'_{24}$ ($x_{25}$) | $x'_{25}$ ($x_3$) | $x'_{26}$ ($x_{29}$) | $x'_{27}$ ($x_5$) | $x'_{28}$ ($x_{23}$) | $x'_{29}$ ($x_{17}$) | $x'_{30}$ ($x_{31}$) | $x'_{31}$ ($x_{33}$) | $x'_{32}$ ($x_1$) | $x'_{33}$ ($x_{19}$) |
| 0.7158 | 0.6857 | 0.6753 | 0.6745 | 0.6733 | 0.6046 | 0.6046 | 0.5995 | 0.5496 | 0.5307 | 0.5147 |

has the shortest distance between the cluster center and the sample. Through clustering algorithm, $N$ cluster centers for the attribute vector $R_k$ can be found, that is to say every cluster center is regarded as a reference vector of $R_k$. As the result, $N$ reference vector of the $k$th attribute vector can be acquired and denoted by $A_k = \{A_1^k, A_2^k, \ldots, A_n^k, \ldots, A_N^k\}$, where $A_n^k$=[$\alpha_{k,n}^1$, $\alpha_{k,n}^2$,..., $\alpha_{k,n}^h$].

**Step 3.2.2: Statistically analyze samples on attribute vectors and cast these samples to the reference vectors.**

Firstly, the relationship between attribute vector $R_k$ and the class $y$ should be transformed into the relationship between the reference vector $\boldsymbol{A}_k$={$A_n^k|n = 1,\ldots, N$} of $R_k$ and the class $y$. $A_n^k$ is initially determined by $k$-means clustering, and then is optimized by a mount of training samples under a specific optimized objective function. For attribute vector $R_k(t)$, it will be compared with every reference vector $A_n^k$, and the similarity distribution that $R_k(t)$ matches $A_n^k$ is as Eq. (16) [21].

$$S_I(R_k(t)) = \{(A_n^k, \beta_{k,n})|k = 1, \ldots, K; n = 1, \ldots, N\} \tag{16}$$

In Eq. (16), $\beta_{k,n}$ represents the similarity that $R_k(t)$ matches the $n$th reference vector $A_n^k$, and is defined by Eq. (17).

$$\beta_{k,n} = \frac{\gamma_n^k}{\sum_{n=1}^N \gamma_n^k} \tag{17a}$$

$$\gamma_n^k = \exp(-\sqrt{(R_k(t) - A_n^k) \times (R_k(t) - A_n^k)^T}) \tag{17b}$$

Eq. (17b) indicates that the shorter distance between attribute vector $R_k(t)$ and reference vector $A_n^k$, the larger similarity that $R_k(t)$ matches $A_n^k$.

All sample pairs $(R_k(t), y(t))$ in dataset $U$ are transformed into the belief distribution $(\beta_{k,1}, \ldots, \beta_{k,N})$ indicating the similarity between sample and reference vectors. Table 1 shows the casting result of samples $(R_k, y)$ on attribute vector $R_k$, where $a_{p,n}$ represents the sum of the integrated similarity that the attribute vector $R_k(t)$ of all samples matches reference vector $A_n^k$ and belongs to class $y_p$.

**Step 3.2.3: REM generation based on likelihood function normalization.**

Based on the casting result shown in Table 1, the belief degree that input attribute vector $R_k(t)$ matches the reference vector $A_n^k$ and the corresponding output $y(t)$ matches the reference value $y_p$ can be acquired by Eq. (18).

$$\mu_{p,n}^k = \frac{a_{p,n}/\delta_p}{\sum_{l=1}^P (a_{l,n}/\delta_l)} \tag{18}$$

In Eq. (18), $\delta_p = \sum_{n=1}^N a_{p,n}$ denotes the sum of the integrated similarity degree of the samples that belong to class $y_p$, and $0 \le \mu_{p,n}^k \le 1$, $\sum_{p=1}^P \mu_{p,n}^k = 1$. The evidence corresponds to the reference vector $A_n^k$ is defined as Eq. (19).

$$e_n^k = [\mu_{1,n}^k, \mu_{2,n}^k, \ldots, \mu_{P,n}^k] \tag{19}$$

Consequently, the evidence matrix describing the relationship between input attribute vector $R_k$ and $N$ classes $y$ can be developed. Table 2 gives the evidence matrix of attribute vector $R_k$.

In Table 2, $\mu_{p,n}^k$ denotes that the belief degree that a sample belongs to $y_p$ given that the attribute vector $R_k$ takes the reference vector $A_n^k$.

### 3.3. Inference process based on ER rule

After the observed values of a set of reference vectors are obtained, the input attribute vector $R_k(t)$ will activate the reference evidence { $e_1^k$,..., $e_n^k$,..., $e_N^k$ } that the reference vector corresponds to in Table 2, and then these pieces of reference evidence are weighting added based on the similarities that $R_k(t)$ matches $A_n^k$ to generate the final evidence $e_k$ corresponding to $R_k(t)$.

$$e_k = \{(y_p, p_{p,k}), p = 1, \ldots, P\} \tag{20a}$$

$$p_{p,k} = \sum_{n=1}^N \beta_{k,n} \mu_{p,n}^k \tag{20b}$$

By using Eq. (20a) and Eq. (20b), we could acquire $K$ pieces of evidence $e_1$,..., $e_k$,..., $e_K$ corresponding to the $K$ input attribute vectors, and set the initial importance weight $w_k$ to be equal to the reliability factor $r_k^R$. $e_1$,..., $e_k$,..., $e_K$ are fused by ER rule as Eq. (5), and the fusion result is

$$O(R(t)) = \{(y_p, p_{p,e(K)}),\ p = 1, \ldots, P\} \tag{21}$$

The class of the sample can be determined which corresponds to the largest belief degree in the fusion result.

### 3.4. Parameters optimization of AV-ER classifier based on genetic algorithm

The parameters of the initial ER classifier include the reference vectors $\boldsymbol{A}_k = \{A_n^k|n = 1, \ldots, N\}$ and the importance weights of evidence $w_k(k = 1, \ldots, K)$. The initial classifier cannot accurately describe the complex mapping relationship between attribute vector $R_k(k = 1, \ldots, K)$ and the class $y_p$, and therefore the classifier parameters should be fine tuned by training dataset $U$ to improve the performance of the AV-ER classifier. The parameters optimization model is as Eq. (22).

$$\min \xi(\boldsymbol{P}) = \sum_{t=1}^T d_E(O(R(t)), V^t) \tag{22a}$$

$$s.t.\quad 0 \le w_k \le 1, \quad k = 1, \ldots, K$$

$$\min(x_{(k-1)h+1}(t)) \le \alpha_{k,n}^1 \le \max(x_{(k-1)h+1}(t))$$

$$\min(x_{(k-1)h+2}(t)) \le \alpha_{k,n}^2 \le \max(x_{(k-1)h+2}(t))$$

**Table 4**
The initial reference vectors of attribute vector $R_1$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^1$ | 0.9305 | 0.8419 | 0.9260 | 0.9260 | 0.9285 | 0.8409 | 0.8532 | 0.9081 |
| $A_2^1$ | −0.7200 | 0.1780 | −0.1044 | −0.2026 | −0.5489 | 0.6099 | 0.3671 | −0.4095 |
| $A_3^1$ | 0.0110 | −0.1434 | 0.2891 | 0.1016 | 0.2049 | −0.2903 | −0.2854 | −0.0588 |
| $A_4^1$ | 0.5548 | 0.3376 | 0.6197 | 0.6304 | 0.5345 | 0.4084 | 0.4957 | 0.5617 |

**Table 5**
The initial reference vectors of attribute vector $R_2$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^2$ | 0.8106 | 0.7781 | 0.8442 | 0.7853 | 0.6915 | 0.7673 | −0.0180 | 0.0169 |
| $A_2^2$ | −0.4902 | −0.2294 | 0.4164 | −0.3900 | 0.1276 | 0.1009 | 0.7413 | 0.7597 |
| $A_3^2$ | 0.3379 | 0.0273 | −0.8078 | −0.7322 | −0.5080 | 0.9254 | 0.4199 | −0.1792 |
| $A_4^2$ | 0.2320 | 0.0095 | 0.3086 | 0.1903 | −0.0534 | 0.4292 | −0.2170 | 0.1167 |

**Table 6**
The initial reference vectors of attribute vector $R_3$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^3$ | 0.6709 | 0.5413 | 0.8624 | 0.2658 | −0.4983 | −0.4547 | 0.7236 | −0.6542 |
| $A_2^3$ | 0.8837 | 0.5938 | 0.4757 | 0.5437 | 0.4975 | 0.6661 | 0.9071 | 0.4527 |
| $A_3^3$ | −0.2977 | −0.1497 | 0.0171 | −0.0325 | −0.2979 | −0.0826 | 0.0712 | 0.0541 |
| $A_4^3$ | 0.8267 | −0.0957 | −0.0504 | −0.0664 | −0.0373 | −0.0244 | 0.7797 | −0.1098 |

**Table 7**
The initial reference vectors of attribute vector $R_4$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $A_1^4$ | 0.1233 | 0.0341 | 0.1795 | −0.1703 | −0.1111 | 0.0478 | 0.1321 | 0.9226 | −0.0843 |
| $A_2^4$ | −0.0316 | −0.3762 | −0.1544 | −0.1137 | −0.2163 | −0.5330 | −0.4419 | 0.9322 | −0.2852 |
| $A_3^4$ | −0.7755 | −0.6710 | 0.7203 | −0.2575 | 0.1554 | 0.7971 | 0.1197 | 0.5294 | −0.6118 |
| $A_4^4$ | 0.1587 | 0.4630 | 0.1134 | 0.3637 | 0.5649 | 0.2983 | 0.2078 | 0.8571 | 0.6572 |

$$\vdots \tag{22b}$$
$$\min(x_{kh}(t)) \le \alpha_{k,n}^h \le \max(x_{kh}(t))$$

In Eq. (22a), $\xi(\boldsymbol{P})$ is the object function of the optimization model, where $d_E$ is the Euclidean distance between the fusion result $O(R(t)) = (p_{1,e(K)}, p_{2,e(K)}, \ldots, p_{P,e(K)})$ (i.e. the vector constituting belief degrees in Eq. (19)) and the reference output vector $V^t$ which is also denoted by belief distribution. The vector of reference belief degrees $V^t$ assigns the absolute belief degree to the real class $y_p$ of sample $R(t)$, for example, for a typical three-class problem, if the real class of $R(t)$ is $y_3$, then $V^t$=(0,0,1). $\boldsymbol{P}$={ $A_n^k$, $w_k|k = 1, \ldots, K; n = 1, \ldots, N\}$ is the parameter set to be optimized. Eq. (22b) lists the constrains that the optimization model should be satisfied.

This paper uses genetic algorithm (GA) as the optimization engine, and according to the criterion that survival of the fittest, the proximate optimal solution can be generated by evolving the initial populations generationally. In every generation, the individuals are selected based on the fitness values of individuals in the parameters set, and then are optimized. New populations represented the new solutions are generated by crossover and variation of genetic operator [24]. With the iterative optimization of populations, the belief degrees in REM as Table 2 reach to the optimal value gradually with the adjustment of reference vectors.

### 3.5. Performance evaluation of classifier based on AIC

Generally, besides classification accuracy, model complexity is also an important indicator to evaluate classifiers. In the classification for high dimensional attributes, the number of model parameters will increase with the rising dimensions of input attributes. Consequently, the classification model becomes complicated and redundant. To solve the problem, Akaike information criterion (AIC) is adopted as the criteria for model evaluation. AIC is a reasonable criterion to measure the model fitness, and can be used to keep balance between the accuracy and complexity of classifiers [25]. Eq. (23) gives the calculation of AIC.

$$AIC = T \times ln(T \times MSE) + 2Num \tag{23}$$

Where $T$ is the sample size of the training dataset, $MSE$ is the mean square error between the model output and the real result, and $Num$ is the number of model parameters. The model is closer to the optimal with a smaller AIC value.

In AV-ER classifier, the number of attribute vectors $K$ has an effect on model parameters $w$ and the number of reference vectors, and further influences model complexity. Hence, Eq. (23) can be used to calculate AIC values under different $K$ value, and we could choose an appropriate $K$ value to develop the best classifier. In order to compare the performance of classifiers under different $K$ value, Eq. (24) gives the relative AIC calculation. In the following classification experiments, Eq. (24) will be used

**Table 8**
The optimal reference vectors of attribute vector $R_1$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^1$ | 0.1154 | −0.8831 | 0.8501 | 0.1793 | 0.3806 | 0.1969 | 0.7506 | 0.8407 |
| $A_2^1$ | −0.3679 | 0.7277 | −0.1481 | −0.4092 | −0.4254 | 0.4150 | −0.3148 | 0.2390 |
| $A_3^1$ | −0.1245 | 0.4169 | 0.5238 | 0.0662 | 0.6671 | 0.0808 | −0.9084 | −0.9916 |
| $A_4^1$ | −0.3557 | 0.6637 | −0.8913 | 0.6399 | −0.0463 | 0.1890 | −0.5934 | −0.3133 |

**Table 9**
The optimal reference vectors of attribute vector $R_2$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^2$ | −0.0115 | −0.2412 | −0.3317 | 0.7326 | 0.5352 | 0.6106 | −0.0990 | 0.3308 |
| $A_2^2$ | 0.4622 | 0.7164 | 0.1855 | 0.0968 | 0.1431 | −0.0392 | −0.0632 | 0.8483 |
| $A_3^2$ | −0.2386 | 0.5434 | −0.6338 | 0.1057 | 0.3494 | −0.1289 | −0.6408 | −0.3894 |
| $A_4^2$ | 0.7770 | 0.5598 | −0.1344 | −0.8988 | 0.4599 | 0.0618 | −0.6507 | −0.8359 |

**Table 10**
The optimal reference vectors of attribute vector $R_3$.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $A_1^3$ | −0.5639 | −0.3467 | 0.0834 | 0.9038 | −0.9125 | −0.0194 | −0.1979 | 0.9468 |
| $A_2^3$ | −0.6685 | −0.6747 | 0.6568 | −0.5755 | 0.4971 | 0.7429 | −0.6416 | 0.2854 |
| $A_3^3$ | −0.6626 | −0.4210 | −0.5193 | 0.6434 | −0.7947 | 0.9633 | 0.9969 | −0.0239 |
| $A_4^3$ | 0.6567 | −0.8430 | −0.8071 | −0.5258 | −0.4504 | −0.6967 | 0.2937 | 0.3149 |

**Table 11**
The optimal reference vectors of attribute vector $R_4$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $A_1^4$ | −0.7430 | −0.7923 | 0.8863 | 0.0570 | −0.5914 | −0.8143 | −0.8088 | 0.7923 | 0.5658 |
| $A_2^4$ | −0.6838 | −0.5889 | −0.4799 | 0.2077 | 0.9265 | 0.5948 | 0.9106 | 0.2422 | 0.7930 |
| $A_3^4$ | 0.8102 | 0.7671 | 0.5180 | 0.3350 | 0.2221 | 0.7158 | 0.8543 | 0.6306 | 0.8395 |
| $A_4^4$ | 0.4602 | −0.7879 | −0.8080 | −0.7236 | −0.5218 | −0.3141 | 0.7289 | 0.9238 | −0.3563 |

**Table 12**
The optimal casting result of ($R_1$,$y$).

| $y$ | $R_1$ | | | | |
|---|---|---|---|---|---|
| | $A_1^1$ | $A_2^1$ | $A_3^1$ | $A_4^1$ | Total |
| $y_1$ | 57.5603 | 63.4936 | 44.6871 | 14.2590 | 180 |
| $y_2$ | 30.8730 | 35.5221 | 17.8399 | 15.7651 | 100 |
| Total | 88.4333 | 99.0157 | 62.5270 | 30.0241 | 280 |

**Table 13**
The optimal casting result of ($R_2$,$y$).

| $y$ | $R_2$ | | | | |
|---|---|---|---|---|---|
| | $A_1^2$ | $A_2^2$ | $A_3^2$ | $A_4^2$ | Total |
| $y_1$ | 43.3955 | 103.4298 | 15.3056 | 17.8690 | 180 |
| $y_2$ | 29.7067 | 39.2819 | 13.4192 | 17.5923 | 100 |
| Total | 73.1022 | 142.7117 | 28.7247 | 35.4613 | 280 |

**Table 14**
The optimal casting result of ($R_3$,$y$).

| $y$ | $R_3$ | | | | |
|---|---|---|---|---|---|
| | $A_1^3$ | $A_2^3$ | $A_3^3$ | $A_4^3$ | Total |
| $y_1$ | 31.4777 | 15.0345 | 17.0479 | 116.4399 | 180 |
| $y_2$ | 20.1395 | 23.6540 | 17.0311 | 39.1754 | 100 |
| Total | 51.6173 | 38.6885 | 34.0790 | 155.6153 | 280 |

**Table 15**
The optimal casting result of ($R_4$,$y$).

| $y$ | $R_4$ | | | | |
|---|---|---|---|---|---|
| | $A_1^4$ | $A_2^4$ | $A_3^4$ | $A_4^4$ | Total |
| $y_1$ | 25.2246 | 14.3826 | 107.0670 | 33.3258 | 180 |
| $y_2$ | 18.0123 | 16.3743 | 42.5085 | 23.1049 | 100 |
| Total | 43.2369 | 30.7569 | 149.5755 | 56.4307 | 280 |

**Table 16**
REM of attribute vector $R_1$.

| $y$ | $R_1$ | | | |
|---|---|---|---|---|
| | $e_1^1$ | $e_2^1$ | $e_3^1$ | $e_4^1$ |
| | $A_1^1$ | $A_2^1$ | $A_3^1$ | $A_4^1$ |
| $y_1$ | 0.5088 | 0.4982 | 0.5819 | 0.3344 |
| $y_2$ | 0.4912 | 0.5018 | 0.4181 | 0.6656 |

**Table 17**
REM of attribute vector $R_2$.

| $y$ | $R_2$ | | | |
|---|---|---|---|---|
| | $e_1^2$ | $e_2^2$ | $e_3^2$ | $e_4^2$ |
| | $A_1^2$ | $A_2^2$ | $A_3^2$ | $A_4^2$ |
| $y_1$ | 0.4480 | 0.5940 | 0.3879 | 0.3607 |
| $y_2$ | 0.5520 | 0.4060 | 0.6121 | 0.6393 |

**Table 18**
REM of attribute vector $R_3$.

| $y$ | $R_3$ | | | |
|---|---|---|---|---|
| | $e_1^3$ | $e_2^3$ | $e_3^3$ | $e_4^3$ |
| | $A_1^3$ | $A_2^3$ | $A_3^3$ | $A_4^3$ |
| $y_1$ | 0.4648 | 0.2610 | 0.3574 | 0.6228 |
| $y_2$ | 0.5352 | 0.7390 | 0.6426 | 0.3772 |

**Table 19**
REM of attribute vector $R_4$.

| $y$ | $R_4$ | | | |
|---|---|---|---|---|
| | $e_1^4$ | $e_2^4$ | $e_3^4$ | $e_4^4$ |
| | $A_1^4$ | $A_2^4$ | $A_3^4$ | $A_4^4$ |
| $y_1$ | 0.4376 | 0.3279 | 0.5832 | 0.4449 |
| $y_2$ | 0.5624 | 0.6721 | 0.4168 | 0.5551 |

to evaluate the performance of AV-ER classifier under different $K$ value, and *cAIC* represents the relative AIC.

$$cAIC_k = \frac{AIC_k}{\max(AIC_k)} \tag{24}$$

## 4. Experiments

In this section, the benchmark in machine learning "Ionosphere" dataset is used to illustrate the design of the AV-ER classifier in detail. Based on AIC, the complexity and accuracy of classifiers under different $K$ values are compared, and the results show that the performance of the AV-ER classifier under a suitable $K$ is superior to that of the ER classifier proposed in Ref. [26]. To further verify the flexibility, the accuracy and complexity of the AV-ER classifier are compared with those of the ER classifier by using another six typical datasets [27,28].

### 4.1. Experiments with Ionosphere dataset

In the Ionosphere dataset, Radar echo of the ionosphere is used to identify whether the ionosphere shows a specific structure. There are 351 samples in the dataset, and each sample has 34 input attributes. Specifically, the radar receives 17 pulse signals, and two features can be extracted from every signals. Consequently, there are 34 attributes in total. Since the values of the samples on one attribute are all zero, only the rest 33 attributes $x_1 \sim x_{33}$ are used as the input attributes. The Ionosphere dataset is on a binary classification, of which the results are "Yes" and "No", and therefore the FoD is $\Theta=\{y_1, y_2\}$. There are 225 samples on $y_1$ class and 126 samples on $y_2$ class.

Five-fold cross validation is conducted on the Ionosphere dataset. Specifically, the whole Ionosphere dataset is divided into five parts, and four parts are used as training dataset while the rest part is used as the testing part in every fold validation. The validation is conducted for five times in turn. Take the first-cross validation as the example to illustrate the model construction. The training dataset is $X=[x_1, \ldots, x_j, \ldots, x_{33}]$, where $x_j=[x_j(1), \ldots, x_j(t), \ldots, x_j(280)]^T$, the rest data samples constitute the testing dataset to verify the model performance in the following.

33 attributes are used to form the attribute vectors with PCA described in Section 3.1. Here, taking $K = 4$ as an example to show how to design the AV-ER classifier. Rank the attributes $x_1 \sim x_{33}$ according to their reliability factors $r$ (Eq. (14b)) in decent order, and renumber them as $x'_1 \sim x'_{33}$. Table 3 lists the attribute numbers before and after the sorting and their corresponding reliability factors.

As $K = 4$, four attribute vectors can be acquired by Eq. (14) which are $R_1=[x'_1, x'_2, \ldots, x'_8]$; $R_2=[x'_9, x'_{10}, \ldots, x'_{16}]$; $R_3=[x'_{17}, x'_{18}, \ldots, x'_{24}]$; $R_4=[x'_{25}, x'_{26}, \ldots, x'_{33}]$. The reliability factors of the four attribute vectors are obtained by Eq. (15), and they are $r_1^R = 0.9600$, $r_2^R = 0.9157$, $r_3^R = 0.7569$, $r_4^R = 0.6030$. After that, rearrange the dataset $X$ in the form of attribute vectors, and the training dataset is $U=\{R(t)=[R_1(t), R_2(t), R_3(t), R_4(t), y(t)] | \, t = 1, 2, \ldots, T, T = 211, y(t) \in \Theta\}$. By using the $k$-means clustering method in Section 3.2, the initial reference vectors of $R_1 \sim R_4$ are listed in Tables 4–7.

By using the training samples, the initial casting results of the attribute vectors and the evidence matrixes can be generated by information transform technology and likelihood function normalization described in Section 3.2. The initial value of importance weight $w_i$ ($i = 1,..,4$) is set as $w_i = r_i^R$. The pieces of evidence activated by any attribute set $[R_1(t), R_2(t), R_3(t), R_4(t)]$ in training dataset $U$ are fused by ER rule, and the category of the attribute set can be determined according to the fusion result.

Finally, the set of optimal parameters $\boldsymbol{P}$ is acquired with the optimization model of AV-ER classifier given in Section 3.4. Tables 8–11 list the reference vectors of $R_1 \sim R_4$ after optimization and cast the training samples based on the optimal reference vectors to generate casting results of $R_1 \sim R_4$ as shown in Tables 12–15. Tables 16–19 give the optimized REMs after likelihood function normalization. Additionally, the best importance weight of $R_1 \sim R_4$ are $w_1 = 0.7854$, $w_2 = 0.4641$, $w_3 = 0.8424$, $w_4 = 0.9449$ respectively.

Finally, the AV-ER classifier optimized by the training dataset can predict any sample in training dataset and testing dataset. The AV-ER classifier developed by the reference vectors in Tables 8–11 can identify the training dataset with the accuracy 0.9607 and the testing dataset with the accuracy 0.9014. However, the classification accuracy of the training dataset and that of the testing dataset are only 0.7964 and 0.7493 respectively by the AV-ER classifier developed by the initial reference vectors in Tables 4–7. It can be seen that the optimization model proposed in Section 3.4 has an significant effect on improving classifier accuracy.

**Table 20**
The results comparison under different $K$ value.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 33 | 0.9117 | 0.8714 | 0.9429 | 0.0245 | 2524.5 | 2401.3 | 2629.2 | 78.04 |
| AV-ER | $K=3$ | 3 | 0.8803 | 0.8286 | 0.9143 | 0.0296 | 2549.8 | 2455.3 | 2650.0 | 66.11 |
| | $K=4$ | 4 | 0.8833 | 0.8429 | 0.9286 | 0.0383 | 2545.0 | 2406.0 | 2627.6 | 101.44 |
| | $K=5$ | 5 | 0.9031 | 0.8857 | 0.9286 | 0.0140 | 2494.4 | 2408.0 | 2540.1 | 44.04 |
| | $\boldsymbol{K=6}$ | **6** | **0.9316** | **0.9000** | **0.9571** | **0.0211** | **2398.6** | **2266.5** | **2504.6** | **86.02** |
| | $K=7$ | 7 | 0.9089 | 0.8873 | 0.9429 | 0.0190 | 2481.4 | 2349.3 | 2544.1 | 67.37 |
| | $K=8$ | 8 | 0.9259 | 0.9000 | 0.9429 | 0.0140 | 2416.4 | 2342.9 | 2499.6 | 50.16 |

**Table 21**
Basic information of eight benchmark datasets.

| No | Name | Total number of samples | The number of classes | The number of attributes | Remarks |
|---|---|---|---|---|---|
| 1 | Australian | 690 | 2 | 14 | Some data is missing in the dataset, and the missing data is filled by random forest regression. |
| 2 | Seeds | 210 | 3 | 7 | / |
| 3 | Hepatitis | 155 | 2 | 19 | Some data is missing in the dataset, and the missing data is filled by random forest regression. |
| 4 | Heart | 270 | 2 | 13 | / |
| 5 | Wine | 178 | 3 | 13 | / |
| 6 | Sonar | 208 | 2 | 60 | / |
| 7 | Dry bean | 13611 | 7 | 16 | / |
| 8 | Musk(version2) | 6598 | 2 | 168 | / |

**Table 22**
The results comparison under different $K$ value on Australian dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 14 | 0.8755 | 0.8273 | 0.8978 | 0.0250 | 5376.3 | 5262.9 | 5559.8 | 103.29 |
| AV-ER | $K=3$ | 3 | 0.7348 | 0.6115 | 0.8489 | 0.1008 | 5781.6 | 5474.9 | 5996.3 | 220.92 |
| | $K=4$ | 4 | 0.8088 | 0.7826 | 0.8540 | 0.0244 | 5614.2 | 5460.7 | 5684.9 | 78.26 |
| | $K=5$ | 5 | 0.7697 | 0.7319 | 0.8613 | 0.0465 | 5696.9 | 5413.1 | 5780.3 | 135.58 |
| | $K=6$ | 6 | 0.8550 | 0.8248 | 0.8841 | 0.0202 | 5464.7 | 5341.3 | 5565.5 | 76.99 |
| | $\boldsymbol{K=7}$ | **7** | **0.8800** | **0.8589** | **0.8968** | **0.0124** | **5319.2** | **5266.3** | **5482.9** | **71.19** |
| | $K=8$ | 8 | 0.7667 | 0.7246 | 0.8623 | 0.0509 | 5732.1 | 5440.3 | 5823.6 | 143.02 |

**Table 23**
The results comparison under different $K$ value on Seeds dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 7 | 0.9079 | 0.8551 | 0.9662 | 0.0388 | 1305.4 | 989.1 | 1558.8 | 191.06 |
| AV-ER | $K=2$ | 2 | 0.8524 | 0.6905 | 0.9524 | 0.0921 | 1313.6 | 1123.5 | 1437.9 | 110.41 |
| | $\boldsymbol{K=3}$ | **3** | **0.9125** | **0.8333** | **0.9762** | **0.0465** | **1285.4** | **1009.1** | **1335.9** | **110.17** |

**Table 24**
The results comparison under different $K$ value on Hepatitis dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 19 | 0.8884 | 0.8467 | 0.9577 | 0.0410 | 988.4 | 830.7 | 1002.6 | 64.77 |
| AV-ER | $K=3$ | 3 | 0.8642 | 0.8333 | 0.8750 | 0.0157 | 984.3 | 978.2 | 1006.0 | 11.16 |
| | $K=4$ | 4 | 0.8203 | 0.6774 | 0.9333 | 0.0867 | 1014.3 | 886.6 | 1086.2 | 67.48 |
| | $K=5$ | 5 | 0.8190 | 0.7742 | 0.8438 | 0.0276 | 1024.2 | 1010.0 | 1052.1 | 17.01 |
| | $\boldsymbol{K=6}$ | **6** | **0.8786** | **0.8125** | **0.9667** | **0.0533** | **962.7** | **798.5** | **1018.9** | **77.79** |
| | $K=7$ | 7 | 0.8457 | 0.7742 | 0.9333 | 0.0508 | 1008.9 | 899.5 | 1056.1 | 52.61 |
| | $K=8$ | 8 | 0.8638 | 0.8065 | 0.9375 | 0.0442 | 986.6 | 894.6 | 1030.9 | 46.19 |

**Table 25**
The results comparison under different $K$ value on Heart dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 13 | 0.8667 | 0.8148 | 0.9259 | 0.0395 | 1792.3 | 1665.4 | 1863.3 | 69.99 |
| AV-ER | $K=3$ | 3 | 0.8074 | 0.7778 | 0.8333 | 0.0189 | 1851.8 | 1820.5 | 1882.7 | 21.05 |
| | $K=4$ | 4 | 0.8111 | 0.7593 | 0.8519 | 0.0319 | 1849.6 | 1797.1 | 1901.9 | 35.97 |
| | $K=5$ | 5 | 0.8444 | 0.7963 | 0.8889 | 0.0343 | 1809.6 | 1736.9 | 1867.9 | 48.19 |
| | $K=6$ | 6 | 0.8519 | 0.7593 | 0.9259 | 0.0586 | 1801.1 | 1651.4 | 1905.9 | 88.19 |
| | $K=7$ | 7 | 0.8296 | 0.7407 | 0.8704 | 0.0459 | 1833.3 | 1774.2 | 1923.9 | 51.63 |
| | $\boldsymbol{K=8}$ | **8** | **0.8630** | **0.8148** | **0.9259** | **0.0432** | **1788.2** | **1655.4** | **1853.3** | **74.28** |

**Table 26**
The results comparison under different $K$ value on Wine dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 13 | 0.9744 | 0.9678 | 0.9835 | 0.0052 | 897.5 | 815.6 | 936.2 | 38.73 |
| AV-ER | $K=3$ | 3 | 0.9214 | 0.8611 | 0.9714 | 0.0412 | 1009.2 | 862.9 | 1091.5 | 81.14 |
| | $K=4$ | 4 | 0.9373 | 0.8108 | 0.9990 | 0.0698 | 969.8 | 802.5 | 1133.0 | 129.36 |
| | $K=5$ | 5 | 0.9685 | 0.9429 | 0.9856 | 0.0140 | 936.0 | 846.6 | 979.4 | 48.13 |
| | $\boldsymbol{K=6}$ | **6** | **0.9713** | **0.9167** | **0.9943** | **0.0283** | **883.3** | **807.3** | **1032.4** | **81.88** |
| | $K=7$ | 7 | 0.9627 | 0.9143 | 0.9963 | 0.0303 | 943.3 | 843.2 | 1034.4 | 73.62 |
| | $K=8$ | 8 | 0.9663 | 0.9444 | 0.9990 | 0.0204 | 898.8 | 802.5 | 971.4 | 65.02 |

Here, a testing sample $R$=[$R_1$,$R_2$,$R_3$,$R_4$] is used to describe the detailed inference process of the AV-ER classifier. The attribute vectors in $R$ are:

$R_1 = [1, 0.9893, 0.9929, 0.9574, 0.9716, 0.9485, 0.9840, 0.9858]$

$R_2 = [0.9787, 0.9449, 1, 0.9716, 0.9911, 0.9754, 0.0195, 0.0160]$

$R_3 = [0.9858, 0.0355, 0.0231, 0.0373, 0.0648, 0.0480, 0.9751, 0.0355]$

$R_4 = [0.0071, 0.0817, 0.0195, 0.0551, 0.0213, 0.0622, 0.0995, 1, 0.0568]$

By using Eq. (17), the similarity between every attribute vector and its corresponding reference vector in Tables 16–19 are calculated. $R_1$ acts evidence $e_1^1 \sim e_4^1$ with the similarity (0.0644, 0.7402, 0.1073, 0.0881), $R_2$ acts evidence $e_1^2 \sim e_4^2$ with the similarity (0.2845, 0.4527, 0.2486, 0.0142), $R_3$ acts evidence $e_1^3 \sim e_4^3$ with the similarity (0.0860, 0.0214, 0.0169, 0.8757), and $R_4$ acts evidence $e_1^4 \sim e_4^4$ with the similarity (0.0109, 0.0467, 0.8956, 0.0467). Then, the evidence of every attribute vector is calculated according to Eq. (20a) which are $e_1$={($y_1$,0.5251), ($y_2$,0.4749)}, $e_2$={($y_1$,0.5108), ($y_2$,0.4892)}, $e_3$={($y_1$,0.5797), ($y_2$,0.4203)}, $e_4$={($y_1$, 0.5292), ($y_2$, 0.4708) }. These pieces of evidence are fused by ER rule with Eq. (5) to generate the fusion result $O(R)$={($y_1$, 0.5965), ($y_2$,0.4035)}, where the belief degree of $y_1$ is larger than that of $y_2$. Consequently, the class of the sample $R$ is determined to be $y_1$, and it is in accordance with the real class of sample $R$.

It is obvious that the number of attribute vectors $K$ significantly influences the performance of classifiers. Institutively, if the $K$ value is larger, the number of attribute vectors is bigger, and more evidence is fused, which means that the information in

**Table 27**
The results comparison under different $K$ value on Sonar dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 60 | 0.8871 | 0.810 | 0.9512 | 0.0434 | 1764.4 | 1621.2 | 1873.3 | 88.28 |
| AV-ER | $K = 5$ | 5 | 0.7172 | 0.6047 | 0.7619 | 0.0597 | 1805.8 | 1778.4 | 1866.4 | 34.27 |
| | $K = 6$ | 6 | 0.7216 | 0.6667 | 0.7805 | 0.0490 | 1813.9 | 1771.5 | 1845.3 | 32.42 |
| | $K = 10$ | 10 | 0.8417 | 0.7857 | 0.9024 | 0.0377 | 1727.7 | 1644.1 | 1779.5 | 44.64 |
| | $K = 12$ | 12 | 0.8037 | 0.6667 | 0.9268 | 0.0990 | 1759.4 | 1592.5 | 1848.2 | 99.02 |
| | $\boldsymbol{K = 15}$ | **15** | **0.8798** | **0.8095** | **0.9286** | **0.0426** | **1691.4** | **1606.1** | **1769.9** | **57.66** |
| | $K = 20$ | 20 | 0.8557 | 0.7619 | 0.9070 | 0.0544 | 1731.8 | 1664.1 | 1817.1 | 59.05 |

**Table 28**
The results comparison under different $K$ value on Dry bean dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 16 | 0.9541 | 0.9498 | 0.9577 | 0.0078 | 9761.4 | 9538.4 | 9977.2 | 171.8 |
| AV-ER | $K = 3$ | 3 | 0.6473 | 0.6129 | 0.6734 | 0.0226 | 11783.3 | 11707.5 | 11876.4 | 61.75 |
| | $K = 4$ | 4 | 0.7644 | 0.5680 | 0.8871 | 0.1180 | 11306.8 | 10665.7 | 12008.6 | 477.49 |
| | $K = 5$ | 5 | 0.8236 | 0.7229 | 0.9231 | 0.0859 | 10957.4 | 10236.3 | 11514.9 | 530.34 |
| | $K = 6$ | 6 | 0.9372 | 0.9153 | 0.9677 | 0.0190 | 10041.6 | 9422.1 | 10382.5 | 362.89 |
| | $\boldsymbol{K = 7}$ | **7** | **0.9525** | **0.9402** | **0.9677** | **0.0111** | **9776.2** | **9413.8** | **10039.2** | **249.43** |
| | $K = 8$ | 8 | 0.9348 | 0.9028 | 0.9516 | 0.0194 | 10086.2 | 9818.8 | 10508.6 | 270.56 |

**Table 29**
The results comparison under different $K$ value on Musk (version 2) dataset.

| Model | | The number of $w$ | Accuracy | | | | *AIC* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Average | Min value | Max value | SD | Average | Min value | Max value | SD |
| ER | | 166 | 0.7417 | 0.6842 | 0.7895 | 0.0422 | 5261.7 | 5187.3 | 5341.4 | 63.80 |
| AV-ER | $K = 15$ | 15 | 0.7500 | 0.6000 | 0.8229 | 0.0889 | 4934.9 | 4823.6 | 5129.2 | 121.19 |
| | $K = 20$ | 20 | 0.8006 | 0.7500 | 0.8511 | 0.0365 | 4870.2 | 4759.8 | 4964.6 | 74.14 |
| | $K = 25$ | 25 | 0.7693 | 0.7579 | 0.8511 | 0.0392 | 4907.9 | 4789.7 | 4979.9 | 79.56 |
| | $K = 30$ | 30 | 0.8130 | 0.8000 | 0.8737 | 0.0364 | 4864.4 | 4721.2 | 4933.9 | 82.79 |
| | $\boldsymbol{K = 50}$ | **50** | **0.8781** | **0.8122** | **0.8958** | **0.0360** | **4837.4** | **4691.9** | **4935.8** | **92.72** |
| | $K = 80$ | 80 | 0.8345 | 0.7668 | 0.8646 | 0.0461 | 4979.4 | 4851.6 | 5100.1 | 89.59 |

attributes can be processed in detail. As a result, the accuracy of classification will increase with the rising $K$ value, but the model complexity will grow which as well prolongs the inference and optimization time. Oppositely, a smaller $K$ value will decrease the parameter number of classifiers, the classification accuracy, and the model complexity. However, the above intuition believes the relationship between $K$ value and the model accuracy or model complexity is linear, but the relationship between input attributes and output classes is significantly nonlinear and uncertain in practical. Hence, the optimal $K$ value should be determined through experiments, which means the model parameters should be less in the premise that the model has a high accuracy. In this condition, the computational burden caused by evidence fusion and model optimization can be controlled, and the model can be in a balanced state.

To achieve a balanced model, we select different $K$ values, and the five-fold cross validation is conducted for every $K$ value. Table 20 lists the *AIC* value of testing datasets in the five-fold cross validation, average classification accuracy (ACA), and the number of importance weight $w$ which should be fine-tuned for the testing datasets. Additionally, the minimum values (Min value), the maximum values (Max value), and the standard deviations (SD) of classification accuracy and AIC in the five-fold cross validation are also described in Table 20, which reflect the stability of the AV-ER model in five-fold cross validation. From Table 20, it can be found that the ACA and *cAIC* can reach a best balance when $K$ equals to 6, which means the evidence to be fused and the importance weight to be optimized will decrease strongly compared with the ER classifier, but the classification accuracy of AV-ER classifier is superior to that of ER classifier.

**Table 30**
ACAs of the seven classifiers on the nine datasets.

| | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | AV-ER |
|---|---|---|---|---|---|---|---|---|---|
| Ionosphere | 0.8829 | 0.9000 | 0.8800 | 0.8914 | **0.9429** | 0.9088 | 0.9088 | 0.9117 | 0.9316 |
| Australian | 0.8014 | 0.7943 | 0.6957 | 0.7246 | 0.8507 | 0.8435 | 0.8638 | 0.8755 | **0.8800** |
| Seeds | 0.8571 | 0.8499 | 0.8000 | 0.8857 | 0.8619 | 0.9038 | 0.8948 | 0.9079 | **0.9125** |
| Hepatitis | 0.5742 | 0.6452 | 0.5548 | 0.4516 | 0.5806 | 0.8517 | 0.8452 | **0.8884** | 0.8786 |
| Heart | 0.7222 | 0.8333 | 0.6519 | 0.7111 | 0.8370 | 0.7815 | 0.8148 | **0.8667** | 0.8630 |
| Wine | 0.9000 | **0.9778** | 0.7296 | 0.8056 | 0.9722 | 0.9238 | 0.9043 | 0.9744 | 0.9713 |
| Sonar | 0.6952 | 0.6333 | 0.7952 | 0.7905 | 0.8048 | 0.7782 | 0.8268 | **0.8871** | 0.8798 |
| Dry bean | 0.8374 | 0.9473 | 0.9080 | 0.9151 | 0.9154 | 0.8543 | 0.8383 | **0.9541** | 0.9525 |
| Musk (V2) | 0.7880 | 0.7396 | 0.8571 | 0.8310 | 0.8446 | 0.8277 | 0.8423 | 0.7417 | **0.8781** |
| Average | 0.7842 | 0.8134 | 0.7635 | 0.7785 | 0.8455 | 0.8548 | 0.8599 | 0.8897 | **0.9052** |

**Table 31**
*cAIC*s of the seven classifiers on the nine datasets.

| | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|
| Ionosphere | 0.9967 | 0.9778 | 1 | 0.9872 | 0.9333 | 0.9486 | 0.9381 | 0.9605 | **0.9127** |
| Australian | 0.8680 | 0.8759 | 1 | 0.9600 | 0.8177 | 0.9871 | 0.9729 | 0.8073 | **0.7987** |
| Seeds | 0.9334 | 0.9413 | 1 | 0.9014 | 0.9282 | 0.8676 | 0.9402 | 0.9122 | **0.8983** |
| Hepatitis | 0.8415 | 0.7895 | 0.8631 | 1 | 0.8346 | 0.9632 | 0.9348 | 0.7711 | **0.7511** |
| Heart | 0.9027 | 0.7823 | 1 | 0.9167 | 0.7789 | 0.9686 | 0.9523 | 0.7653 | **0.7636** |
| Wine | 0.8107 | **0.7462** | 1 | 0.9057 | 0.7505 | 0.8510 | 0.9061 | 0.7590 | 0.7471 |
| Sonar | 0.9110 | 1 | 0.7964 | 0.8011 | 0.7869 | 0.9647 | 0.9501 | 0.8105 | **0.7770** |
| Dry bean | 0.9654 | 0.9847 | 0.9778 | **0.9556** | 0.9634 | 0.9845 | 0.9933 | 0.9784 | 0.9739 |
| Musk (V2) | 0.9820 | 0.9649 | 0.9779 | 0.9731 | 0.9884 | 0.9648 | **0.9553** | 0.9851 | 0.9801 |
| Average | 0.9123 | 0.8954 | 0.9572 | 0.9334 | 0.8646 | 0.9444 | 0.9492 | 0.8610 | **0.8447** |

**Table 32**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Ionosphere.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.8918 | 0.9079 | 0.8876 | 0.8991 | 0.9451 | 0.9142 | 0.9159 | 0.9201 | **0.9462** |
| | SDCA | 0.0314 | 0.0376 | 0.0284 | 0.0302 | 0.0318 | 0.0337 | 0.0388 | 0.0326 | **0.0280** |
| 30% | ACA | 0.8879 | 0.9058 | 0.8834 | 0.8957 | **0.9439** | 0.9105 | 0.9129 | 0.9165 | 0.9405 |
| | SDCA | 0.0301 | 0.0326 | 0.0297 | 0.0365 | 0.0288 | 0.0333 | 0.0355 | 0.0319 | **0.0275** |
| 40% | ACA | 0.8840 | 0.9008 | 0.8832 | 0.8889 | **0.9404** | 0.9041 | 0.9015 | 0.9145 | 0.9356 |
| | SDCA | 0.0294 | 0.0304 | 0.0287 | 0.0334 | **0.0261** | 0.0321 | 0.0331 | 0.0298 | 0.0264 |
| 50% | ACA | 0.8778 | 0.8921 | 0.8711 | 0.8794 | 0.9369 | 0.9005 | 0.8972 | 0.9089 | **0.9289** |
| | SDCA | 0.0288 | 0.0312 | 0.0296 | 0.0329 | 0.0247 | 0.0322 | 0.0318 | 0.0304 | **0.0241** |

### *4.2. Experiments on eight benchmark datasets*

To further verify the flexibility and effectiveness of the AV-ER classifier, this model will be applied in eight benchmark datasets in UCI database for machine learning which are Hepatitis dataset, Heart dataset, Sonar dataset, Australian dataset, Seeds dataset, Wine dataset, Musk (version 2) dataset, and dry bean dataset. Table 21 lists the basic information of these datasets. In detail, the Hepatitis dataset is to predict whether a patient has hepatitis according to the relevant diagnostic indexes; the Heart dataset is to predict whether a patient has a heart disease, and the attributes in the dataset are general personal information and some relevant test results; the Sonar dataset is to identify the target is rock or mine by using the signal strength of Sonar return from the targets in different angles; the Australian dataset is on credit card application, which include eight attributes and six

**Table 33**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Australian.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.8112 | 0.7998 | 0.7054 | 0.7315 | 0.8624 | 0.8521 | 0.8719 | 0.8818 | **0.8867** |
| | SDCA | 0.0342 | 0.0412 | 0.0370 | 0.0312 | 0.0297 | 0.0344 | 0.0319 | **0.0261** | 0.0266 |
| 30% | ACA | 0.8082 | 0.7966 | 0.7012 | 0.7281 | 0.8567 | 0.8460 | 0.8694 | 0.8787 | **0.8844** |
| | SDCA | 0.0326 | 0.0374 | 0.0357 | 0.0323 | 0.0315 | 0.0328 | 0.0342 | 0.0264 | **0.0242** |
| 40% | ACA | 0.8041 | 0.7910 | 0.6941 | 0.7278 | 0.8495 | 0.8471 | 0.8677 | 0.8734 | **0.8828** |
| | SDCA | 0.0311 | 0.0355 | 0.0365 | 0.0347 | 0.0312 | 0.0333 | 0.0319 | 0.0271 | **0.0234** |
| 50% | ACA | 0.7979 | 0.7887 | 0.6919 | 0.7208 | 0.8465 | 0.8344 | 0.8626 | 0.8708 | **0.8769** |
| | SDCA | 0.0316 | 0.0311 | 0.0352 | 0.0319 | 0.0334 | 0.0347 | 0.0317 | 0.0262 | **0.0221** |

**Table 34**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Seeds.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.8612 | 0.8575 | 0.8122 | 0.8954 | 0.8704 | 0.9189 | 0.9040 | 0.9169 | **0.9188** |
| | SDCA | 0.0227 | 0.0241 | 0.0250 | 0.0324 | 0.0257 | 0.0197 | 0.0218 | 0.0221 | **0.0191** |
| 30% | ACA | 0.8585 | 0.8546 | 0.8076 | 0.8887 | 0.8671 | 0.9154 | 0.9001 | 0.9137 | **0.9160** |
| | SDCA | 0.0214 | 0.0225 | 0.0287 | 0.0319 | 0.0247 | 0.0262 | 0.0227 | **0.0182** | 0.0187 |
| 40% | ACA | 0.8545 | 0.8527 | 0.8025 | 0.8837 | 0.8619 | 0.9087 | 0.8977 | 0.9107 | **0.9128** |
| | SDCA | 0.0234 | 0.0241 | 0.0301 | 0.0334 | 0.0267 | 0.0228 | 0.0245 | 0.0189 | **0.0166** |
| 50% | ACA | 0.8519 | 0.8441 | 0.7995 | 0.8789 | 0.8585 | 0.9005 | 0.8901 | 0.9042 | **0.9100** |
| | SDCA | 0.0262 | 0.0249 | 0.0284 | 0.0307 | 0.0301 | 0.0289 | 0.0252 | 0.0182 | **0.0175** |

**Table 35**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Hepatitis.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.5920 | 0.6556 | 0.5645 | 0.4677 | 0.5934 | 0.8631 | 0.8571 | **0.8945** | 0.8838 |
| | SDCA | 0.0244 | 0.0227 | 0.0227 | 0.0334 | 0.0279 | 0.0191 | 0.0252 | 0.0179 | **0.0131** |
| 30% | ACA | 0.5865 | 0.6502 | 0.5597 | 0.4631 | 0.5895 | 0.8597 | 0.8546 | **0.8913** | 0.8797 |
| | SDCA | 0.0268 | 0.0261 | 0.0189 | 0.0308 | 0.0261 | 0.0239 | 0.0277 | **0.0138** | 0.0144 |
| 40% | ACA | 0.5788 | 0.6503 | 0.5534 | 0.4582 | 0.5859 | 0.8564 | 0.8500 | **0.8864** | 0.8713 |
| | SDCA | 0.0231 | 0.0208 | 0.0164 | 0.0287 | 0.0289 | 0.0192 | 0.0274 | 0.0160 | **0.0157** |
| 50% | ACA | 0.5682 | 0.6411 | 0.5507 | 0.4479 | 0.5775 | 0.8488 | 0.8445 | **0.8855** | 0.8724 |
| | SDCA | 0.0217 | 0.0227 | **0.0140** | 0.0240 | 0.0234 | 0.0165 | 0.0248 | 0.0146 | 0.0164 |

numerical labels; the Seeds dataset is to determine the categories of wheat seeds; the Wine dataset provides the chemical analysis of three kinds of wine; the dry bean dataset uses 16 features to describe seven dry beans types; the musk (version 2) dataset describes a set of 102 molecules of which 39 are judged by human experts to be musks and the remaining 63 molecules are judged to be non-musks. Table 21 lists the detailed information of the eight datasets.

Similarly, five-fold cross-validation was performed on the above datasets and Tables 22–29 show the comparisons between the AV-ER classifier and the ER classifier on the eight datasets respectively.

As shown by the above experiment results, the classification accuracies of the AV-ER classifier on Australian dataset ($K = 7$), Seeds dataset ($K = 3$), and Musk (version 2) dataset ($K = 50$) are higher than those of ER classifier, while the classification accuracies of the AV-ER classifier on Hepatitis dataset ($K = 6$), Heart dataset ($K = 7$), Wine dataset ($K = 6$), Sonar dataset ($K = 15$) and Dry bean dataset ($K = 7$) are slightly lower than those of ER classifier. However, for all the datasets except the dry bean dataset, the model complexity of the AV-ER classifier is obviously smaller than that of ER classifier. The AIC of AV-ER classifier for dry bean dataset is quite similar with that of ER classifier for the dataset. In Tables 22–29, the number of attribute

**Table 36**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Heart.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.7344 | 0.8419 | 0.6611 | 0.7188 | 0.8469 | 0.7880 | 0.8215 | 0.8746 | **0.8769** |
| | SDCA | 0.0289 | 0.0245 | 0.0339 | 0.0353 | 0.0288 | 0.0430 | 0.0311 | **0.0281** | 0.0289 |
| 30% | ACA | 0.7302 | 0.8378 | 0.6556 | 0.7154 | 0.8421 | 0.7827 | 0.8184 | 0.8691 | **0.8705** |
| | SDCA | 0.0261 | 0.0230 | 0.0267 | 0.0310 | 0.0257 | 0.0366 | 0.0273 | 0.0255 | **0.0246** |
| 40% | ACA | 0.7275 | 0.8283 | 0.6586 | 0.7074 | 0.8357 | 0.7791 | 0.8104 | 0.8614 | **0.8641** |
| | SDCA | 0.0231 | 0.0269 | 0.0226 | 0.0323 | 0.0231 | 0.0318 | 0.0246 | **0.0228** | 0.0230 |
| 50% | ACA | 0.7203 | 0.8167 | 0.6478 | 0.7031 | 0.8319 | 0.7767 | 0.8086 | 0.8585 | **0.8589** |
| | SDCA | 0.0219 | 0.0222 | 0.0204 | 0.0267 | 0.0219 | 0.0274 | 0.0221 | 0.201 | **0.186** |

**Table 37**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Wine.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.9069 | 0.9821 | 0.7379 | 0.8125 | 0.9788 | 0.9284 | 0.9123 | 0.9791 | **0.9825** |
| | SDCA | 0.0326 | 0.0359 | 0.0325 | 0.0397 | 0.0334 | 0.0366 | 0.0402 | 0.0334 | **0.0300** |
| 30% | ACA | 0.9027 | **0.9777** | 0.7318 | 0.8071 | 0.9741 | 0.9247 | 0.9087 | 0.9751 | 0.9769 |
| | SDCA | 0.0297 | 0.0311 | 0.0285 | 0.0355 | 0.0289 | 0.0321 | 0.0358 | 0.0308 | **0.0276** |
| 40% | ACA | 0.8975 | 0.9728 | 0.7251 | 0.8014 | 0.9687 | 0.9194 | 0.9011 | 0.9711 | 0.9732 |
| | SDCA | 0.0264 | 0.0256 | 0.0263 | 0.0327 | 0.0260 | 0.0288 | 0.0349 | 0.0299 | **0.0237** |
| 50% | ACA | 0.8924 | **0.9700** | 0.7210 | 0.8000 | 0.9650 | 0.9166 | 0.8981 | 0.9665 | 0.9680 |
| | SDCA | 0.0233 | 0.0227 | 0.0255 | 0.0304 | 0.0230 | 0.0256 | 0.0327 | 0.0239 | **0.0219** |

**Table 38**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Sonar.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.7025 | 0.6425 | 0.8037 | 0.8013 | 0.8113 | 0.7885 | 0.8325 | 0.8931 | **0.8943** |
| | SDCA | 0.0315 | 0.0326 | 0.0364 | 0.0311 | 0.0289 | 0.0337 | 0.0268 | 0.0254 | **0.0233** |
| 30% | ACA | 0.6970 | 0.6389 | 0.7983 | 0.7986 | 0.8076 | 0.7857 | 0.8286 | 0.8865 | **0.8871** |
| | SDCA | 0.0285 | 0.0337 | 0.0351 | 0.0288 | 0.0267 | 0.0314 | 0.0251 | 0.0242 | **0.0217** |
| 40% | ACA | 0.6929 | 0.6365 | 0.7940 | 0.7940 | 0.8011 | 0.7814 | 0.8250 | 0.8820 | **0.8826** |
| | SDCA | 0.0272 | 0.0313 | 0.0324 | 0.0266 | 0.0246 | 0.0288 | 0.0235 | **0.0224** | 0.0240 |
| 50% | ACA | 0.6904 | 0.6305 | 0.7911 | 0.7869 | 0.7985 | 0.7767 | 0.8207 | **0.8781** | 0.8750 |
| | SDCA | 0.0264 | 0.0300 | 0.0304 | 0.0251 | 0.0255 | 0.0274 | 0.0221 | **0.0216** | 0.0231 |

vectors influences the performance of AV-ER classifier both in accuracy and AIC. The AV-ER classifier can perform well with an appropriate $K$ value. In the five-fold cross validation, for most datasets, the differences between minimum value and maximum value of accuracy are not obvious, and the standard deviations are below 0.05 except for that of the Hepatitis dataset. The AIC values in the five-fold cross validation also vary slightly which are all below 100 except dry bean dataset and the seed dataset. Overall, AIC can assist the decision maker to determine the appropriate parameters of the AV-ER classifier and keep balance between classification accuracy and model complexity.

### *4.3. Comparisons with some mainstream classifiers on nine benchmark datasets*

To further verify the effectiveness of the ER classifier, the AV-ER classifier are compared with eight typical classifiers which are decision tree, naïve Bayse, k-nearest neighbor algorithm, SVM, random forest, BP neural network (BPNN), ensemble learning, and ER classifier. Similarly, hyperparametric optimization and five-fold cross validation are also conducted on the nine datasets by using every typical algorithm respectively. The hyperparameters

**Table 39**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Dry bean.

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.8445 | 0.9548 | 0.9148 | 0.9200 | 0.9221 | 0.8612 | 0.8441 | 0.9582 | **0.9594** |
| | SDCA | 0.0289 | 0.0277 | 0.0324 | 0.0313 | 0.0387 | 0.0298 | 0.0327 | 0.0264 | **0.0240** |
| 30% | ACA | 0.8395 | 0.9526 | 0.9105 | 0.9164 | 0.9182 | 0.8587 | 0.8411 | **0.9564** | 0.9560 |
| | SDCA | 0.0264 | 0.0252 | 0.0311 | 0.0303 | 0.0376 | 0.0277 | 0.0296 | 0.0230 | **0.0235** |
| 40% | ACA | 0.8352 | 0.9490 | 0.9052 | 0.9129 | 0.9159 | 0.8561 | 0.8357 | **0.9522** | 0.9538 |
| | SDCA | 0.0274 | 0.0264 | 0.0281 | 0.0279 | 0.0334 | 0.0252 | 0.0305 | 0.0237 | **0.0220** |
| 50% | ACA | 0.8316 | 0.9438 | 0.9017 | 0.9104 | 0.9126 | 0.8515 | 0.8314 | 0.9491 | **0.9510** |
| | SDCA | 0.0254 | 0.0239 | 0.0266 | 0.0258 | 0.0310 | 0.0228 | 0.0278 | 0.0225 | **0.0202** |

**Table 40**
The statistical analyses of ACA&SDCA for different percentages of test samples (PT) in dataset Musk (V2).

| PT | SI | Decision tree | Naïve Bayse | k-nearest neighbor | SVM | Random forest | BPNN | Ensemble learning | ER | **AV-ER** |
|---|---|---|---|---|---|---|---|---|---|---|
| 20% | ACA | 0.7994 | 0.7486 | 0.8650 | 0.8381 | 0.8527 | 0.8346 | 0.8500 | 0.7494 | **0.8859** |
| | SDCA | 0.0354 | 0.0352 | 0.0411 | 0.0337 | 0.0294 | 0.0343 | 0.0316 | **0.0252** | 0.0267 |
| 30% | ACA | 0.7924 | 0.7451 | 0.8615 | 0.8349 | 0.8502 | 0.8312 | 0.8467 | 0.7460 | **0.8824** |
| | SDCA | 0.0317 | 0.0313 | 0.0386 | 0.0306 | 0.0277 | 0.0282 | 0.0323 | 0.0267 | **0.0241** |
| 40% | ACA | 0.7861 | 0.7405 | 0.8580 | 0.8279 | 0.8479 | 0.8250 | 0.8386 | 0.7410 | **0.8790** |
| | SDCA | 0.0323 | 0.0283 | 0.0308 | 0.0264 | 0.0259 | 0.0252 | 0.0301 | 0.0244 | **0.0238** |
| 50% | ACA | 0.7830 | 0.7362 | 0.8519 | 0.8251 | 0.8405 | 0.8226 | 0.8347 | 0.7356 | **0.8761** |
| | SDCA | 0.0288 | 0.0301 | 0.0269 | 0.0240 | 0.0267 | 0.0239 | 0.0304 | 0.0231 | **0.0220** |

selected by eight classifiers are detailed in Appendix C. The performance of every classifier on the nine datasets is shown in Tables 30 and 31.

From Tables 30 and 31, it can be found that AV-ER classifier has outstanding classification accuracy. Although the accuracy of AV-ER classifier is lower than random forest on Sonar dataset, lower than ER classifier on Hepatitis dataset, Heart dataset and dry bean dataset, lower than Naïve Bayse on Wine dataset, the classification accuracy of AV-ER classifier is always in top three, and it has the best average accuracy. AV-ER classifier well keeps the balance between model complexity and accuracy and has the smallest *cAIC* value on most dataset except Wine dataset, dry bean dataset, and Musk (Version 2) dataset. AV-ER classifier is also has the smallest average *cAIC* value.

### *4.4. The refined statistical analysis of classification experiment results*

In the above five-fold cross-validation experiment, the percentage of test samples (PT) is only set as 20% of the total samples. In order to test for the stability of the AV-ER, here we further choose more cases (PT = 20%,30%,40%,50%) in respective 100 times random experiments of nine datasets classification to verify the performance of AV-ER. In Table 32 to Table 40, the above seven classifiers and ER classifier are selected to compare with the AV-ER classifier in different PT cases.

Here two main statistical indices (SI) of the nine classifiers are calculated including ACA and the standard deviation of classification accuracy (SDCA). SDCA describes the deviation between the mean value of classification accuracy and single classification accuracy obtained in one random experiment. The small value of SDCA means the corresponding classifier can provide relative stable performance in every test, namely there is rarely overly-well or overly-poor classification result. From Tables 32–40, it can be seen that AV-ER provides a relatively small SDCA compared with the other methods, meanwhile its ACA is always in the top three in all PT cases. This shows that AV-ER has high classification performance and algorithmic stability. Certainly, with the increase of PT from 20% to 50%, the percentage of training samples (PT) decreases for 80% to 50%, respectively. Moreover, the number of training samples for modeling these nine classifiers also decreases at the same time, which causes their ACAs are all reduce gradually.

## 5. Conclusions

This paper proposes a classifier based on attribute vectorization and evidential reasoning. Firstly, form the attribute vectors with the attribute variables, and determine the reliability factors of the attribute vectors. Then, generate the REMs by likelihood function normalization with the training dataset, and fuse the activated evidence by ER rule to make the classification decision. In the optimization process, fine tune the parameters of the initial classifier to improve the model performance, and use AIC to comprehensively evaluate the complexity and accuracy of the classifier. Finally, choose the typical benchmark datasets in the UCI database to verify the effectiveness and flexibility of the proposed method compared with the ER classifier.

**Table A.1**
Notations.

| | | | |
|---|---|---|---|
| ER | Evidential reasoning | $T$ | The total numbers of samples in sample set |
| AV-ER | Attribute vectorization based evidential reasoning | $K$ | The total numbers of attribute vectors |
| PCA | Principal component analysis | $X$ | The dataset containing $T$ samples with $J$ attributes |
| AIC | Akaike information criterion | $F_i$ | The $i$th principal component after PCA |
| SVM | Support vector machine | $w_{ji}$ | The weighting coefficient of the $j$th attribute to the $i$th principal component |
| DS | Dempster–Shafer | $W$ | Transformation matrix composed of weighting coefficients |
| FoD | A framework of discernment | $\widetilde{X}$ | The matrix after centralization |
| BBA | Basic belief assignment | $V$ | the correlation coefficient matrix of $\widetilde{X}$ |
| REM | Reference evidence matrix | $\lambda$ | Eigenvalues of $V$ |
| *MSE* | Mean square error | $\omega$ | Eigenvector of $V$ |
| *Num* | The number of model parameters | $\varphi_j$ | The contribution of the $j$th principal component |
| *cAIC* | The relative AIC | $M_D$ | The cumulative variance contribution in PCA |
| ACA | Average classification accuracy | $\overline{\varphi}_j$ | The importance of the $j$th attribute to all principal components |
| SD | Standard deviations | $R_k$ | Attribute vector |
| $\Theta$ | the framework of discernment | $r_k^R$ | The reliability factor of $R_k$ |
| $P(\Theta)$ | The power set of $\Theta$ | $\boldsymbol{A}_k$ | The reference vector for $R_k$ |
| $x_j$ | The $j$th attribute extracted from observed samples | $A_n^k$ | The $n$th reference vector in $\boldsymbol{A}_k$ |
| $y_p$ | The sample belongs to the $p$th class | $\beta_{k,n}$ | The similarity that $R_k(t)$ matches the $n$th reference vector $A_n^k$ |
| $\theta$ | A proposition in $\Theta$ | $a_{p,n}$ | The sum of the integrated similarity that $R_k(t)$ of all samples matches $A_n^k$ and belongs to $y_p$ |
| $e_j$ | The $j$th piece of evidence supporting $\theta$ | $\mu_{p,n}^k$ | The belief degree that $R_k(t)$ matches $A_n^k$ |
| $r_j$ | Reliability factor of evidence $e_j$ | $e_n^k$ | Evidence corresponds to $A_n^k$ |
| $w_j$ | Importance weight of evidence $e_j$ | $e_k$ | The final evidence corresponding to $R_k$ |
| $p_{\theta,j}$ | The belief degree that $e_j$ supporting $\theta$ | $\xi(\boldsymbol{P})$ | Objective function in parameters optimization |
| $\widetilde{m}_{\theta,j}$ | The supporting degree of $e_j$ to $\theta$ considering $r_j$ and $w_j$ | $P$ | Parameters to be optimized |
| $m_{\theta j}$ | The supporting degree of $e_j$ to $\theta$ considering $w_j$ | $V^t$ | Reference output vector |
| $p_{\theta,e(2)}$ | The belief degree of evidence $e_1$ and $e_2$ jointly supporting $\theta$ | | |

The new method inherits the advantages of the ER classifier on dealing with uncertain information fusion, and further solves the problems on model complexity and computation burden caused by high dimensional input attributes. The superiorities of the AV-ER classifier are as follows: (1) In the process of evidence acquisition, the REMs are generated based on the casting results of the attribute vectors. A newly obtained sample will activate all pieces of evidence in the REMs that the reference attribute

**Table B.1**
PCA pseudocode.

***Algorithm* 1 *Attribute vectorization base on PCA.***

***Input***: *The sample set contains T samples ,and each sample has J attribute:* $X=\{x_1,x_2,...,x_J\}$

***OutPut***: *K attribute vectors* $\{R_1,R_2,...,R_k,...,R_K\}$, *The reliability factory of the* $R_k$ : $r_k^R$

***step*1: *J attribute → the principal components space.***

$F=\{F_1,F_2,...,F_J\} \leftarrow XW$

***step*2: *Select the first D principle components***

1-*sample data centralization*

$\tilde{x}_j \leftarrow x_j - u_j$

2-*thie correlation coefficient matrix:*

$V \leftarrow \tilde{X} \leftarrow \{\tilde{x}_1,\tilde{x}_2,...,\tilde{x}_j\}$

3-*eigenvalue decomposition for V:*

$\lambda_J$ *and* $w_j \leftarrow V$

4-*the contributionn of* $F_j$:

$\varphi_j \leftarrow \lambda_j / \sum_{j=1}^{J} \lambda_j$

5-*the cumulative variance contribution:*

$M_D \leftarrow \sum_{i=1}^{D} \lambda_j / \sum_{j=1}^{J} \lambda_j$

6-*the extracted D principal components:*

$F_D \leftarrow \tilde{X} W_D \, (D \leftarrow M_D \geq 0.9)$

7-*the reliability factor of the* $j_{th}$ *attribute*

$r_j \leftarrow \overline{\varphi} \,/\, max(\overline{\varphi})$ $(\overline{\varphi} \leftarrow \sum_{i=1}^{D} \varphi w_{ji})$

***step*3: *Attribute vectorization base on attribute importance***

1-*rank attribute base on* $r_j$:

$sort(r_j) \rightarrow x_1,x_2,...,x_J$

2-*generate K Sub-vectors:*

$\{R_1,R_2,...,R_k,...,R_K\}$

3-*calculate the reliability factor*

$r_k^R = 1/h \sum_{j=(k-1)h+1}^{kh} r_j$

vectors correspond to so that the acquired evidence can contain more useful information. (2) Divide the high dimensional attributes into multiple attribute vectors according to the importance ranking of every attribute, which reduces the number of input attribute and the number of importance weight *w*. With this method, the model complexity decreases, while the model accuracy can be ensured, in order that the optimal balance between classification accuracy and model complexity can be achieved.

## CRediT authorship contribution statement

**Xiaojian Xu:** Draft writing, Model design, Verification, Review comments response. **Xiaobin Xu:** Model construction, Verification. **Pengfei Shi:** Experiment conduction to verify the effectiveness of the model and the review comments response. **Zifa Ye:** Draft writing, Experiment conduction to verify the effectiveness of the model. **Yu Bai:** Comparative analysis of experimental results, Model effectiveness verification. **Shuo Zhang:** Comparative analysis of experimental results. **Schahram Dustdar:** Model design, modification of manuscript. **Guodong Wang:** Model design, modification of manuscript.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

We acknowledge financial support from the National Nature Science Foundation of China (No. 61903108), the Natural Science Foundation of Zhejiang Province, China (LY21F030011), Zhejiang Province Outstanding Youth Fund (LR21F030001), the Zhejiang Province Key R&D projects (No. 2019C03104, 2021C03015), Zhejiang Province Public Welfare Technology Application Research Project (No. LGF20H270004, No. LGF19H180018), Key Project of Zhejiang Provincial Medical and Health Science and Technology Plan (WKJ-ZJ-2038).

## Appendix A

See Table A.1.

**Table B.2**
ER rule and GA pseudocodes.

***Algorithm* 2 *inference process based on ER rule.***

***Input***：$U=\{R(t)=[R_1(t),R_2(t),...,R_k(t),...,R_K(t)],y(t) \mid t=1,2,...,T,y(t)\in\{y_1,y_2,...,y_p,...,y_P\}\}$

***OutPut***：*Classification results:* $O(R(t))$

***step*1:** ***Determine the reference vector of every attribute vector by k-means clusering.***

$A_k^n=\{\alpha_{n,k}^1,\alpha_{n,k}^2,...,\alpha_{n,k}^h\} \leftarrow$ *N clustercenters* $\leftarrow$ *attribute vector* $R_k$ *by k-means*

$A_k=\{A_1^k,A_2^k,...,A_n^k,...,A_N^k\}$

***step*2:** ***Cast thesse samples to the reference vectors.***

1-*thie similarity distribution that* $R_k(t)$ *matches:*

$\gamma_k^n=exp(-\sqrt{(R_k(t)-A_n^k)\times(R_k(t)-A_n^k)^T})$

$\beta_{k,n}=r_k^n \,/\, \sum_{n=1}^{N} r_k^n$

$S_I(R_k(t))=\{(A_k^n,\beta_{k,n})\}$

2-*Casting result of samples* $(R_k,y)$ *on attribute vector* $R_k$

→ ***table*1**

***step*3:** ***REM generation based on likelihood fuction normalization.***

1-*output* $y(t)$ *matches the reference value* $y_p$

$\delta_P=\sum_{n=1}^{N}\alpha_{p,n}$

*when* $R_k(t)=A_k^n$, $u_{n,p}^k=(\alpha_{p,n}\,/\,\delta_P)\,/\,\sum_{l=1}^{P}(\alpha_{l,n}\,/\,\delta_l)$

2-*define the evidence corresponding to reference vector A:*

$e_n^k=[(u_{1,n})^k,(u_{2,n})^k,...,(u_{P,n})^k]$

3-*REM for input attribute vector* $R_k$*:*

→ ***table*2**

***step*4:** ***Inference process.***

1-*reference evidence activation:*

*input: aset of attribute vector* $R_k(t)$

*activate the reference evidence:* $e_{1k},...,e_{nk},...,e_{Nk}$

2-*acquire K pieces of evidence:*

$\{e_1,...,e_k,...,e_K\} \leftarrow e_k=\{(y_p,P_{p,k}),p=1,...,p,...,P\}$

3-*the evidence is fused by ER rule to obtain the classification resules:*

*initial importance weight:* $w_k=r_k^R$

*the fused evidence:* $e(J)=\{(\theta,p_{\theta,e(J)})\}$

*the fusion resule:* $O(R(t))=\{(y_p,p_{p,e(K)}),p=1,...,p,...,P\}$

*the classification result :* $y_p$*,where* $max(P_{p,e(K)})$

***step*5:** ***Parameters optimization of AV-ER classifier based on genetic algorithm.***

*input* → *setofparameterstobeoptimized:*

$P=\{A_n^k,w_k \mid k=1,...,K;n=1,...,N\}$

*object function:*

$\xi(P)=\sum_{t=1}^{T} d_E(O(R(t)),V^t)$

*output* → *Optimized parameter set P* → ***step4***

***step*6:** ***Performance evaluation of classifier based on AIC.***

$AIC=T\times ln(T\times MSE)+2Num$

$CAIC_k=AIC_k\,/\,max(AIC_k)$

**Table C.1**
Hyperparametric information of classical classification methods.

| | | Ionosphere | Australian | Seeds | Heart | Sonar | Dry bean | Musk (V2) |
|---|---|---|---|---|---|---|---|---|
| Decision tree | MinLeafSize | 8 | 5 | 5 | 9 | 9 | 6 | 1 |
| Naïve Bayse | Data distributions | kernel | Gaussian | Gaussian | Gaussian | kernel | Gaussian | kernel |
| k-nearest neighbor | Width | 0.3624 | – | 2.0711 | – | 0.0871 | – | 3.6932 |
| | Distance | cityblock | seuclidean | seuclidean | seuclidean | spearman | mahalanobis | seuclidean |
| | NumNeighbors | 1 | 25 | 1 | 8 | 1 | 1 | 14 |
| SVM | C | 12.757 | 21.876 | 0.9055 | 242.06 | 0.1105 | 49.248 | 0.0015 |
| | KernelScale | 1.5007 | 60.232 | 0.0085 | 6.5783 | 0.0761 | 117.21 | 10.186 |
| Random forest | NumPredictorsToSample | 6 | 4 | 3 | 8 | 8 | 4 | 13 |
| BPNN | MinLeafSize | 2 | 5 | 2 | 4 | 1 | 3 | 7 |
| | Hidden layer | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| | Neuron node | 22 | 8 | 31 | [26 33] | [20 18] | [40 35] | [45 80] |
| Ensemble learning | Method | LogitBoost | AdaBoostM1 | AdaBoostM2 | LogitBoost | AdaBoostM1 | AdaBoostM2 | AdaBoostM1 |
| | NumLearningCycles | 500 | 16 | 10 | 33 | 177 | 14 | 76 |
| | LearnRate | 0.0739 | 0.2702 | 0.6606 | 0.6679 | 0.9865 | 0.9021 | 0.8406 |
| | MinLeafSize | 4 | 24 | 4 | 3 | 1 | 3 | 1 |

**Table C.2**
Description of parameter interpretation.

| | | |
|---|---|---|
| Decision tree | MinLeafSize | Minimum number of leaf node observations. |
| Naïve Bayse | Data distributions | Kernel smoothing density estimate/Multinomial distribution/Multivariate multinomial distribution/Normal (Gaussian) distribution. |
| | Width | Kernel smoothing window width. (optional when distribution 'kernel') |
| k-nearest neighbor | Distance | Distance Metric Names.(cityblock/chebychev/correlation/cosine/euclidean/euclidean/ euclidean/mahalanobis/minkowski/minkowski/spearman) |
| | NumNeighbors | Number of nearest neighbors to find. |
| SVM | C | Penalty factor. |
| | KernelScale | Kernel scale parameter. The random basis of random feature extension is obtained by using kernel scale parameters. |
| Random forest | NumPredictorsToSample | Number of variables to select at random for each decision split. |
| | MinLeafSize | Minimum number of observations per tree leaf. |
| BPNN | Hidden layer | Number of hidden layers. |
| | Neuron node | Number of neuron nodes in each hidden layer. |
| Ensemble learning | Method | Ensemble aggregation method . (Bag/Subspace/AdaBoostM1/AdaBoostM2/GentleBoost/LogitBoost/LogitBoost/ RobustBoost/RUSBoost/TotalBoost) |
| | NumLearningCycles | Number of ensemble learning cycles. |
| | LearnRate | Learning rate for shrinkage. |
| | MinLeafSize | Minimum number of observations per tree leaf. |

## Appendix B

See Tables B.1 and B.2.

## Appendix C

See Tables C.1 and C.2.

## References


[1] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley-interscience, New York, 2000.

[2] Q. Huang, Y. Chen, L. Liu, D. Tao, X. Li, On combining biclustering mining and AdaBoost for breast tumor classification, IEEE Trans. Knowl. Data Eng. 32 (4) (2020) 728–738, http://dx.doi.org/10.1109/TKDE.2019.2891622.

[3] F. Nie, L. Tian, R. Wang, X. Li, Multiview semi-supervised learning model for image classification, IEEE Trans. Knowl. Data Eng. 32 (12) (2020) 2389–2400, http://dx.doi.org/10.1109/TKDE.2019.2920985.

[4] R. Toma, A. Prosvirin, J. Kim, Bearing fault diagnosis of induction motors using a genetic algorithm and machine learning classifiers, Sensors (Basel, Switzerland) 20 (7) (2020).

[5] R. Xu, M. He, Application of deep learning neural network in online supply chain financial credit risk assessment, in: 2020 International Conference on Computer Information and Big Data Applications (CIBDA) 2020.

[6] P. Sornsuwit, S. Jaiyen, A new hybrid machine learning for cybersecurity threat detection based on adaptive boosting, Appl. Artif. Intell. (2019) 1–21.

[7] R.A. Amjad, B.C. Geiger, Learning representations for neural network-based classification using the information bottleneck principle, IEEE Trans. Pattern Anal. Mach. Intell. 42 (9) (2020) 2225–2239, http://dx.doi.org/10.1109/TPAMI.2019.2909031.

[8] M. Zadkarami, A. Safavi, M. Taheri, F. Salimi, Data driven leakage diagnosis for oil pipelines: An integrated approach of factor analysisl and deep neural network classifier, Trans. Inst. Meas. Control 42 (14) (2020) 014233122092814.

[9] J. Xu, J. Han, F. Nie, X. Li, Multi-view scaling support vector machines for classification and feature selection, IEEE Trans. Knowl. Data Eng. 32 (7) (2020) 1419–1430, http://dx.doi.org/10.1109/TKDE.2019.2904256.

[10] Wei, He, Yigang, et al., A naive-Bayes-based fault diagnosis approach for analog circuit by using image-oriented feature extraction and selection technique, IEEE Access (2019) 5065–5079.

[11] C. Nunes, H. Langet, M. De Craene, O. Camara, B. Bijnens, A. Jonsson, Decision tree learning for uncertain clinical measurements, IEEE Trans. Knowl. Data Eng. 33 (9) (2021) 3199–3211, http://dx.doi.org/10.1109/TKDE.2020.2967378.

[12] A. Algarni, Automated medical diagnosis system based on multi-modality image fusion and deep learning, Wirel. Pers. Commun. 111 (4) (2020).

[13] Z. Yu, L. Chang, B. Qian, A belief-rule-based model for information fusion with insufficient multi-sensor data and domain knowledge using evolutionary algorithms with operator recommendations, Soft Comput. 23 (13) (2019) 5129–5142.

[14] D. Li, Y. Deng, K.H. Cheong, Multisource basic probability assignment fusion based on information quality, Int. J. Intell. Syst. (2021).

[15] S. Zhong, X. Liu, A new method to determine basic probability assignment based on interval number, Comput. Commun. IoT Appl. (2019).

[16] B. Qin, F. Xiao, A non-parametric method to determine basic probability assignment based on kernel density estimation, IEEE Access (2018) 1.

[17] X.B. Xu, J. Zheng, J.B. Yang, D.L. Xu, Y.W. Chen, Data classification using evidence reasoning rule, Knowl. Based Syst. 116 (2017) 144–151.

[18] X. Xu, D. Zhang, Y.Bai, L. Chang, Evidence reasoning rule-based classifier with uncertainty quantification, Inform. Sci. 516 (2019).

[19] X. Xu, Z. Zhao, X. Xu, et al., Machine learning-based wear fault diagnosis for marine diesel engine by fusing multiple data-driven models, Knowl.-Based Syst. 190 (2019) 105324.

[20] L.L. Chang, Y. Zhou, J. Jiang, M.J. Li, X.H. Zhang, Structure learning for belief rule base expert system: A comparative study, Knowl.-Based Syst. 39 (2016) 159–172.

[21] J.B. Yang, D.L. Xu, Evidential reasoning rule for evidence combination, Artif. Intell. 205 (205) (2013) 1–29.

[22] P. Comon, Independent component analysis, a new concept? Signal Process. 36 (3) (1994) 287–314.

[23] D.L. Xu, J.B. Yang, Y.M. Wang, The evidential reasoning approach for multi-attribute decision analysis under interval uncertainty, European J. Oper. Res. 174 (3) (2006) 1914–1943.

[24] X. Zan, Z. Wu, C. Guo, Z. Yu, A Pareto-based genetic algorithm for multi-objective scheduling of automated manufacturing systems, Adv. Mech. Eng. 12 (1) (2020) 168781401988529.

[25] L. Chang, Z. Zhou, Y. Chen, X. Xu, J. Sun, T. Liao, X. Tan, Akaike information criterion-based conjunctive belief rule base learning for complex system modelling, Knowl.-Based Syst. 161 (2018) 47–64.

[26] X.B. Xu, J. Zheng, J.B. Yang, D.L. Xu, Y.W. Chen, Data classification using evidence reasoning rule, Knowl.-Based Syst. 116 (2016).

[27] K. Bache, M. Lichman, Uci machine learning repository, 2013.

[28] UCI Machine Learning Repository: http://archive.ics.uci.edu/ml/index.php.