PLOS ONE

RESEARCH ARTICLE

Rough sets and Laplacian score based cost-sensitive feature selection

Shenglong Yu[1,2], **Hong Zhao**[1,2]*

1 Fujian Key Laboratory of Granular Computing and Application (Minnan Normal University), Zhangzhou, Fujian, China, **2** Key Laboratory of Data Science and Intelligence Application, Fujian Province University, Zhangzhou, Fujian, China

* hongzhaocn@163.com

Citation: Yu S, Zhao H (2018) Rough sets and Laplacian score based cost-sensitive feature selection. PLoS ONE 13(6): e0197564. https://doi.org/10.1371/journal.pone.0197564

Editor: Quan Zou, Tianjin University, CHINA

Received: September 24, 2017

Accepted: December 10, 2017

Published: June 18, 2018

Data Availability Statement: The data underlying this study have been uploaded to Github and are accessible using the following link: https://github.com/fhqxa/PLOSONE-D-17-34607. In addition, we have given all datasets and codes.

Funding: This work was supported by the National Natural Science Foundation of China under Grant No.61703196, and the Natural Science Foundation of Fujian Province under Grant No.2018J01549. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Abstract

Cost-sensitive feature selection learning is an important preprocessing step in machine learning and data mining. Recently, most existing cost-sensitive feature selection algorithms are heuristic algorithms, which evaluate the importance of each feature individually and select features one by one. Obviously, these algorithms do not consider the relationship among features. In this paper, we propose a new algorithm for minimal cost feature selection called the rough sets and Laplacian score based cost-sensitive feature selection. The importance of each feature is evaluated by both rough sets and Laplacian score. Compared with heuristic algorithms, the proposed algorithm takes into consideration the relationship among features with locality preservation of Laplacian score. We select a feature subset with maximal feature importance and minimal cost when cost is undertaken in parallel, where the cost is given by three different distributions to simulate different applications. Different from existing cost-sensitive feature selection algorithms, our algorithm simultaneously selects out a predetermined number of "good" features. Extensive experimental results show that the approach is efficient and able to effectively obtain the minimum cost subset. In addition, the results of our method are more promising than the results of other cost-sensitive feature selection algorithms.

Introduction

Feature selection [1–4] is an essential process for machine learning applications [5–7], because it improves generalization capabilities and reduces running time [8–10]. The goal of the feature selection problem is to find a feature subset to reduce the dimensionality of the feature space and improve the predictive accuracy of a classification algorithm [11–16]. There are various feature evaluation methods such as maximal margin [17], maximal stability [18], effective distance [19], maximum relevance-maximum significance [20], and matrix factorization subspace learning [21, 22]. These evaluation methods assume that the obtained data are free. However, in many real-world applications, we should pay test costs for collecting data items [23–25]. Test costs are often measured by time, money, and other resources [26]. Therefore, the cost must be considered in the feature selection process.

Cost-sensitive feature selection (CSFS) [27–29] focuses on selecting a feature subset with the minimal cost as well as one that preserves a particular property of the decision system [30–32]. The CSFS problem becomes a feature selection problem when the cost of CSFS problem is zero. Thus, the CSFS problem is more generalization than the feature selection problem, and it has attracted a lot of research interest recently. The main aim of CSFS algorithms is to search for the cheapest feature subset that preserves sufficient information for classification and clustering (see, e.g., [31, 33, 34]).

In recent years, there have been many work on cost-sensitive feature selection. Tan proposed cost-sensitive feature selection and used it in robotics [35]. Zhang proposed a cost-sensitive feature selection with respect to waiting cost [36]. Min used basic concepts of rough set theory to propose cost-sensitive attribute reduction [31]. Cost-sensitive feature selection based on decision-theoretic was proposed by Jia [37]. Zhao considered numerical data with measurement errors and proposed a cost-sensitive feature selection algorithm based on backtracking approach [38]. Yang proposed a backtracking algorithm for granular structure selection with minimal test cost in [39]. The Semi-greedy heuristics for cost-sensitive feature selection was proposed by Min in [40]. However, these heuristic algorithms evaluate each feature individually and select features one by one. They do not consider the relationship among features and have high time complexity.

In this paper, we propose a cost-sensitive feature selection algorithm based on Rough sets and Laplacian score (CSFS-RSLS) to address the CSFS by considering the trade-off between feature score and test cost. We aim to select a feature subset with maximal feature importance and minimal cost. Thus, each feature is evaluated by both Laplacian score and test cost. Laplacian score can evaluate features according to their locality preserving ability. The cost is given by three different distributions [31, 41] to simulate different applications. It is distinguished from the existing heuristic algorithms, the proposed algorithm takes into account the relationship among features and simultaneously selects out a predetermined number of "good" features.

Nine open datasets from the University of California-Irvine (UCI) library are employed to study the performance of our algorithm. The proposed algorithm is implemented in our open resource software called COst-SEnsitive Rough sets (COSER) [42]. The experimental results show that CSFS-RSLS can select an optimal feature subset with the best exponential weight setting. Compared with two heuristic algorithms [43, 44], CSFS-RSLS algorithm provides an efficient solution on eight datasets. In addition, our algorithm significantly reduces the time complexity and resource consumption.

In subsequent sections, we firstly presents the test-cost-sensitive decision system. Secondly, the subsection includes two points, one is the Laplacian score with cost sensitive, the other is our proposed algorithm. The CSFS problem in our algorithm is defined in this subsection. Subsequently, we show two evaluation metrics, which can evaluate the performance of our proposed CSFS-RSLS algorithm. Fourthly, we discuss the experiment process and list some settings and results. Finally, we provide conclusion and future work.

Test-cost-sensitive decision system

In real applications, we consider decision system with test cost. The test-cost-sensitive decision system is a fundamental concept in data mining and machine learning. The first part shows the test-cost-sensitive decision system models. There are a number of measurement methods with different test costs to obtain a numerical data item. We define test-cost-sensitive decision system with error ranges in the second part.

Definition 1 [45]. A test-cost-sensitive decision system (TCS-DS) S is the 6-tuple:

$$S = (U, C, D, \{V_a | a \in C \cup D\}, \{I_a | a \in C \cup D\}, c^*), \quad (1)$$

where U is a finite set of objects called the universal, C is the set of conditional features, D is the set of decision features. For each $a \in C \cup D$, I_a: $U \rightarrow V_a$. V_a is the set of values for each $a \in C \cup D$, and I_a is an information function for each feature $a \in C \cup D$, $c^* : 2^C \rightarrow \mathbb{R}^+ \cup \{0\}$ is the feature subset test cost function, where $\mathbb{R}^+$ is the set of positive real numbers.

We assume that the test sequence does not influence the total cost, and for each $A \subseteq C$, one should specify the value of $c^*(A)$. Therefore, the feature subset test cost function c^* is to employ a vector

$$c^* = [c^*(\emptyset), c^*(\{a_1\}), c^*(\{a_2\}), \cdots, c^*(\{a_1, a_2\}), \cdots, c^*(C)]. \quad (2)$$

The space requirement for storing function c^* is $2^{|C|}$, which soon becomes unacceptable as $|C|$ increases. To deal with this problem, we need to develop an alternative representation of the test cost function. Therefore, we set $c : C \rightarrow \mathbb{R}^+ \cup \{0\}$ is the test cost function and

$$c^*(A) = \sum_{a \in A} c^*(\{a\}) = \sum_{a \in A} c(a), \quad (3)$$

we assume that the range of cost function c is non-negative $(\mathbb{R}^+ \cup \{0\})$, which is a natural assumption in reality. A feature test cost function can easily be represented by a vector $c = [c(a_1), c(a_2), \cdots, c(a_{|C|})]$.

Definition 2 [43]. Let $S = (U, C, D, V, I, c^*, e)$ be a TCS-DS-ER, where U, C, D, V, I and c^* have the same meaning as Definition 1, $e : C \rightarrow \mathbb{R}^+ \cup \{0\}$ is the maximal error range of $a \in C$, and $\pm e(a)$ is the error range of a. The error range of feature a is defined as

$$e(a) = \Delta \frac{\sum_{i=1}^{m} a(x_i)}{m}, \quad (4)$$

where we set $\Delta = 0.1$, $a(x_i)$ is the i-th instance value of $a \in C$, $i \in [1, m]$, and m is the number of instances. The precision of $e(a)$ can be adjusted through Δ setting.

In order to facilitate processing and comparison, the conditional feature values are normalized, and their value range from 0 to 1. In fact, there are a number of normalization approaches. We employ a simple function of normalization: $y = (x - min)/(max - min)$, where y is the normalized value, x is the initial value, and max and min are the maximal and minimal values in each conditional features.

Table 1 presents a decision system of *Bupa liver disorder* (*Liver* for short), which conditional features are normalized values; where $U = \{x_1, x_2, \cdots, x_{345}\}$, C = {Mcv, Alkphos, Sgpt, Sgot, Gammagt, Drinks}, and D = {Selector}. Table 2 presents an example of test cost vector.

Rough sets and Laplacian score based cost-sensitive feature selection

In this section, we introduce the relative reduct by Rough sets, the Laplacian score (LS) and our algorithm. The first part describes the relative reduct in numeric data. The second part describes the use of LS in cost-sensitive feature selection. Our CSFS-RSLS algorithm is described in the last part. The key of the exponential weighting algorithm is the feature importance exponent weighted function, test costs, and a user-specified exponent α.

Table 1. A numerical decision system (*Liver*).

Patient	Mcv	Alkphos	Sgpt	Sgot	Gammagt	Selector
x_1	0.53	0.60	0.27	0.29	0.09	1
x_2	0.53	0.36	0.36	0.35	0.06	2
x_3	0.55	0.27	0.19	0.14	0.17	2
x_4	0.68	0.48	0.20	0.25	0.11	2
x_5	0.58	0.41	0.05	0.30	0.02	2
x_6	0.87	0.28	0.06	0.16	0.04	2
x_7	0.61	0.34	0.11	0.16	0.01	2
...	...	...	...	...	...	...
x_{344}	0.68	0.39	0.15	0.27	0.03	1
x_{345}	0.87	0.66	0.35	0.52	0.21	1

https://doi.org/10.1371/journal.pone.0197564.t001

Relative reducts in rough sets

Rough set theory [46], proposed in the early 1980s, is a mathematical tool to deal with uncertainty and is a relatively new soft computing method. Concept of relative reduct has been thoroughly investigated by the rough set theory. The concept of relative reduct is built on decision systems, and there are many different definitions, such as positive approximation reducts [47], parallel reducts [48, 49], and a general definition reducts [50].

Definition 3. Any $B \subseteq C$ is a decision-relative reduct if $POS_B(D) = POS_C(D)$, and $\forall a \in B$, $POS_{B-\{a\}}(D) \subset POS_B(D)$.

The first condition guarantees that the information in terms of the positive region is preserved, and the second condition guarantees that no superfluous test is included. With this decision-relative reduct, decision-relative core is naturally defined as follows.

Definition 4. Let $Red(S)$ denotes the set of all decision-relative reducts of S. The decision-relative core of S is $Core(S) = \cap Red(S)$.

In other words, $Core(S)$ contains those tests appearing in all decision-relative reducts. A decision-relative reduct is also called a reduct for brevity.

Laplacian score in cost-sensitive feature selection

In real-world applications, the LS can be applied to supervised or unsupervised feature selection. For many datasets, the local structure of the space is more important than the global structure. To represent the local geometry of the data, LS is used to construct a nearest-neighbor graph. The nearest-neighbor graph of the LS is based on the observation that, two data points are probably related to the same topic if they are close to each other. This can be well-preserved under the data structure and the nearest neighbor graph can be obtained. The importance of each feature is calculated from the nearest neighbor graph. For each feature, the basic idea of LS is to evaluate the feature importance according to its locality preserving power. Here, we apply the LS to unsupervised feature selection.

Table 2. An example of test cost vector.

a	Mcv	Alkphos	Sgpt	Sgot	Gammagt
$c(a)$	\$16.00	\$20.00	\$45.00	\$28.00	\$33.00

https://doi.org/10.1371/journal.pone.0197564.t002

For each feature, we assume that the feature importance is $LS(a)$, $a \in C$. We combine feature importance and cost as follows:

$$LS(a, c) = LS(a)c(a)^{\alpha}, \tag{5}$$

where α is a user-specified non-positive exponent and c is the cost of feature. If $\alpha = 0$, this function reduces to the traditional feature importance. About function $LS(a)$ [51], let's set some symbols. Let $LS(a_r)$ denote the Laplacian Score of the r-th feature. Let f_{ri} denote the i-th sample of the r-th feature, $i = 1, \cdots, m$. Our function can be stated as the following four main steps:

1. Construct a nearest neighbor graph G with m samples. The i-th sample corresponds to x_i. We put an edge between samples i and j if x_i and x_j are "close", i.e. x_i is among k nearest neighbors of x_j or x_j is among k nearest neighbors of x_i. When the label information is available, one can put an edge between two nodes sharing the same label.

2. If samples i and j are connected, put $S_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$, where t is a suitable constant. Otherwise, put $S_{ij} = 0$. The weight matrix S of the graph models the local structure of the data space.

3. For the r-th feature, we define $\mathbf{f}_r = [f_{r1}, f_{r2}, \cdots, f_{rm}]^T$, $D = diag(S\mathbf{1})$, $\mathbf{1} = [1, \cdots, 1]$, and $L = D - S$, where the maxtrix L is often called graph Laplacian. Let

$$\tilde{\mathbf{f}}_r = \mathbf{f}_r - \frac{\mathbf{f}_r^T D\mathbf{1}}{\mathbf{1}^T D\mathbf{1}}\mathbf{1}. \tag{6}$$

4. Compute the LS of the r-th feature as follow:

$$LS(a_r) = \frac{\tilde{\mathbf{f}}_r^T L\tilde{\mathbf{f}}}{\tilde{\mathbf{f}}_r^T D\tilde{\mathbf{f}}}. \tag{7}$$

Example 1 Firstly, we use a subtable of Table 1 as shown in Table 3 and obtain an error range vector in Table 4 by Table 3. Secondly, we obtain the core feature is Gammagt by Table 4 and set $k = 3$ and $t = 1$, the weight matrix S can be written as follow:

$$S = \begin{pmatrix} 1 & 0 & 0 & 0.81 & 0.80 & 0 \\ 0 & 1 & 0.77 & 0.86 & 0 & 0 \\ 0 & 0.77 & 1 & 0.76 & 0 & 0 \\ 0.81 & 0.86 & 0 & 1 & 0 & 0 \\ 0.80 & 0 & 0 & 0.81 & 1 & 0 \\ 0 & 0.75 & 0 & 0.73 & 0 & 1 \end{pmatrix}.$$

Table 3. A subtable of the *Liver* decision system.

Patient	Mcv	Alkphos	Sgpt	Sgot	Gammagt	Selector
x_1	0.53	0.60	0.27	0.29	0.09	1
x_2	0.68	0.36	0.20	0.35	0.12	2
x_3	0.55	0.27	0.19	0.14	0.17	2
x_4	0.68	0.48	0.20	0.25	0.11	1
x_5	0.58	0.57	0.05	0.30	0.08	2
x_6	0.87	0.28	0.06	0.20	0.09	1

https://doi.org/10.1371/journal.pone.0197564.t003

Table 4. An error range vector.

a	Mcv	Alkphos	Sgpt	Sgot	Gammagt
$e(a)$	0.06	0.04	0.02	0.03	0.01

https://doi.org/10.1371/journal.pone.0197564.t004

Then, for each feature, the Laplacian Score $LS(a)$ is shown in Table 5.

The value of the $LS(a)$ indicates the quality of the feature. Table 5 shows that the feature importance is **Gammagt** > **Sgpt** > **Sgot** > **Mcv** > **Alkphos**. When we add the cost and set $\alpha = -1$, the $LS(Mcv, c) = LS(Mcv)c(Mcv)^{-1} = 0.8456 \times 16^{-1} = 0.052$; $LS(Alkphos, c) = 0.037$; $LS(Sgpt, c) = 0.021$; $LS(Sgot, c) = 0.033$; $LS(Gammagt, c) = 0.029$; Obviously, after considering the feature importance, the cost and the core. We choose the order is: **Gammagt** > **Mcv** > **Alkphos** > **Sgot** > **Sgpt**. As opposed to considering only feature importance or cost, the result is very different.

The proposed algorithm

To more quickly and efficiently deal with the problem of test cost, we propose a feature-importance function that includes cost sensitivity to calculate the feature score. This function combines feature importance and cost, and is more reasonable and more widely applicable to practical problems. The algorithm pseudocode is listed in Algorithm 1 and contains two main steps:

1. Add the core feature to B according to reduct of rough sets;
2. Add the current-best feature to B according to feature importance function $LS(a, c)$ until the number of B set achieve the desired number of features.

Algorithm 1 Rough sets and Laplacian score based cost-sensitive feature selection (CSFS-RSLS).

```
Input: (U, C, D, {V_a|a ∈ C ∪ D}, {I_a|a ∈ C ∪ D}, c)
Output: A feature subset with minimal test cost
Method: CSFS-RSLS
1:  B = ∅;
    //Core computing
2:  for (i = 1; i ≤ |C|; i + +)do
3:    if (POS_{C − {a_i}}(D) ≠ POS_C(D))then
4:       B ← B ∪ {a_i}; //{a_i} is a core feature
5:    end if
6:  end for
7:  For any a ∈ C, compute LS(a, c); //compute the laplacian score for
    each feature
    //Addition feature
8:  CA = C − B;
9:  Select d features a ∈ CA with the maximal LS(a, c);
10: B ∪ {a};
11: return B;
```

Table 5. A feature importance vector of the *Liver* subtable.

a	Mcv	Alkphos	Sgpt	Sgot	Gammagt
$LS(a)$	0.8456	0.7439	0.9506	0.9345	0.9680

https://doi.org/10.1371/journal.pone.0197564.t005

Lines 7 and 9 are key to this algorithm. In line 7, we can insert different feature importance significance functions $LS(a, c)$ to obtain various algorithms. Line 9, where d is determined by the comparison algorithm, selects the feature subset number. Eq (4) is introduced here to adjust the influence of the test cost. We use a comparison method in the proposed algorithm to choose the best features subset for different α values.

Algorithm 1 has the following three main advantages over existing algorithms:

1. Computation time is reduced. Because the time complexity of the backtracking algorithm for finding a dataset reduction is $2^{|C|}$, where $|C|$ is the feature number of the dataset, when $|C|$ is large, the calculation time is impractically high. Algorithm 1 shows that the time complexity of the CSFS-RSLS algorithm is $|C|$.
2. It can handle large datasets, which a number of existing algorithms cannot handle. Large datasets have a very large number of features, and existing algorithms would find it very difficult to operate on them. However, in our algorithm, because the time complexity of the CSFS-RSLS algorithm is $|C|$, the CSFS-RSLS algorithm is suitable for large datasets.
3. It is highly likely to produce feature subsets with the minimal total cost. For instance, in Table 2, the existing algorithm obtains the feature subset {Mac, Alkphos, Gammagt}, which has a total test cost is \$16 + \$20 + \$33 = \$69. Our algorithm obtains the feature subset fMac, Alkphos, Sgotg, and its total test cost is \$16 + \$20 + \$28 = \$64. Obviously, our algorithm is better.

Evaluation method

Among the existing algorithms, there are many algorithms to deal with the MTR problem. It is necessary to define several evaluation methods to compare the performances. First, we need a method to evaluate the quality of one feature subset. For example, if the test cost of the optimal feature subset is \$100, an equal number of feature subsets with test cost \$120 are better than another with a test cost of \$150. We propose algorithm can run on many datasets or one dataset with different test cost settings. We propose two statistical metrics: average below factor and average exceeding factor.

Below factor

For a dataset produce test cost setting, let R' be an optimal reduct. The below factor of a feature subset R is

$$bf(R) = \frac{c^*(R') - c^*(R)}{c^*(R')}. \tag{8}$$

The below factor is a quantitative metric for evaluating the performance of a feature subset. It shows the goodness of a feature subset when it is better than the optimal. Naturally, if R is an optimal feature subset, the below factor is 0.

Maximal below factor. To demonstrate the performance of the algorithm, statistical metrics are needed. Let the number of experiments be K. In the i-th experiment ($1 \leq i \leq K$), the feature subset computed by the algorithm is denoted by R_i. The maximal below factor (MBF) is defined as

$$\max_{1 \leq i \leq K} bf(R_i). \tag{9}$$

This is the best case of the algorithm given the dataset. To some extent, it can express the performance of this algorithm.

Average below factor. The average below factor (ABF) is defined as

$$\frac{\sum_{i=1}^{K_1} bf(R_i)}{K_1}. \tag{10}$$

Because ABF is averaged over K_1 different test cost settings, the value of K_1 is $c^*(R)$ less than $c^*(R')$. This value is a very good way to show the performance of the algorithm from a solely statistical perspective.

Exceeding factor

For a dataset with a test cost setting, the exceeding factor is used to show the performance of the algorithm. Similarly, if the algorithm is run K times, the exceeding factor and the maximal exceeding factor are defined in [31]. The exceeding factor provides a quantitative metric to evaluate the performance of a feature subset. It shows the badness of a feature subset when it is not optimal. The value of the maximal exceeding factor is the worst case for some datasets. Although it relates to the performance of one particular feature subset, it should be viewed as a statistical rather than individual metric.

The average exceeding factor (AEF) is defined as

$$\frac{\sum_{i=1}^{K_2} ef(R_i)}{K_2}. \tag{11}$$

The maximal exceeding factor is averaged on $K_2 = K - K_1$ different test cost settings. It is a statistical metric that represents the overall performance of the algorithm.

Experiments

In this section, we try to answer the following questions by experimentation.

1. Is the running time of our algorithm reduced?
2. Is our algorithm efficient?
3. Is the CSFS-RSLS algorithm effective?
4. Is our algorithm appropriate for the minimal test cost feature selection problem?
5. Is there an optimal setting of α that is valid for any dataset?

Datasets

Nine standard datasets are used to study the efficiency and effectiveness of the proposed CSFS-RSLS algorithm. The nine standard datasets of Machine Learning Databases are: *Liver*, *Wpbc*, *Promoters*, *Voting*, *Ionosphere*, *Credit*, *Prostate-GE*, *SMK-CAN-187*, and *Waveform*. The *SMK-CAN-187* [52] is a benchmark microarray based gene expression database and it has 187 samples and 19993 features. The other 8 datasets are from the UCI [53] library. Where *Liver* and *Wpbc* datasets are from medical applications. The *Promoters* dataset is from game applications, *Voting* dataset is from society applications, *Ionosphere* dataset is from physics applications, and *Credit* dataset is from commerce applications. *Prostate-GE* dataset has 102 samples and 5966 features from medical applications, the *Waveform* dataset has 5000 samples and 40 features from Vocality applications.

Table 6. Datasets information.

No.	Name	Domain	$\|U\|$	$\|C\|$	$\|D\|$
1	Liver	Clinic	345	6	2
2	Wpbc	Clinic	198	33	2
3	Promoters	Game	106	57	2
4	Voting	Society	435	16	2
5	Ionosphere	Physics	351	34	2
6	Credit-g	Commerce	1000	20	2
7	Waveform	Vocality	5000	40	3
8	Prostate-GE	Clinic	102	5966	2
9	SMK-CAN-187	Society	187	19993	2

https://doi.org/10.1371/journal.pone.0197564.t006

The data of our experiments come from real applications. However, because these datasets do not provide the test cost, we use uniform, normal, and pareto distributions to generate random test costs in [1, 100]. To help show the performance of the cost-sensitive feature selection algorithm, we create these data for the experiments. The data underlying this study have been uploaded to Github and are accessible using the following link: https://github.com/fhqxa/PLOSONE-D-17-34607.

Their basic information are listed in Table 6, where $|C|$ is the number of features, and $|U|$ is the number of instances, $|D|$ is the number of classes.

Comparison of three distributions

For each dataset, we have different α values, and there are three distributions for generating the test cost settings. The algorithm is run 100 times with different test cost settings and different α settings on nine datasets.

Figs 1–9 show the results of finding the optimal factors from the three distributions. The proposed algorithm performs the best with the pareto distribution for each dataset. Except for the *Ionosphere* dataset, the normal distribution leads to the worst performance. A possible reason is that the pareto distribution generates many small values and a few large values, and there are many features with both low test costs and large LSs. In contrast, the normal

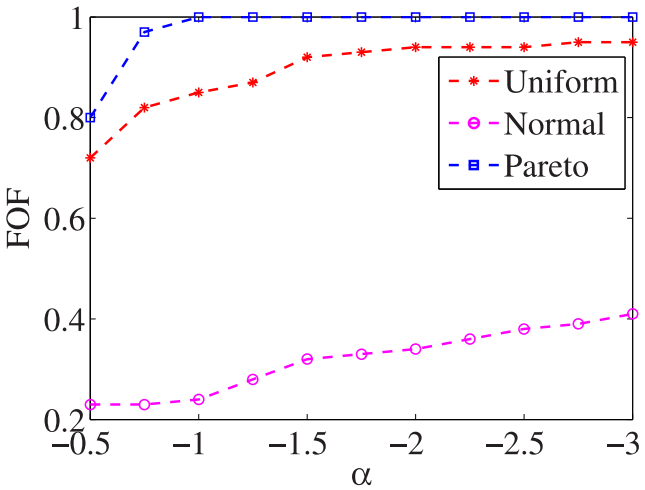

Fig 1. Finding optimal factor of Liver dataset.

https://doi.org/10.1371/journal.pone.0197564.g001

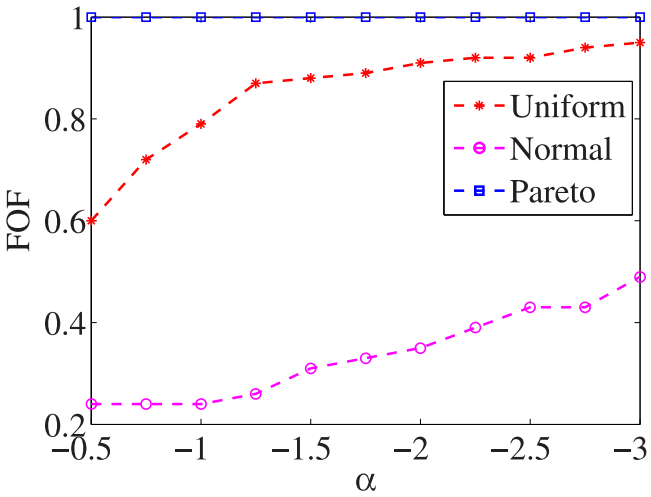

Fig 2. Finding optimal factor of Wpbc dataset.

https://doi.org/10.1371/journal.pone.0197564.g002

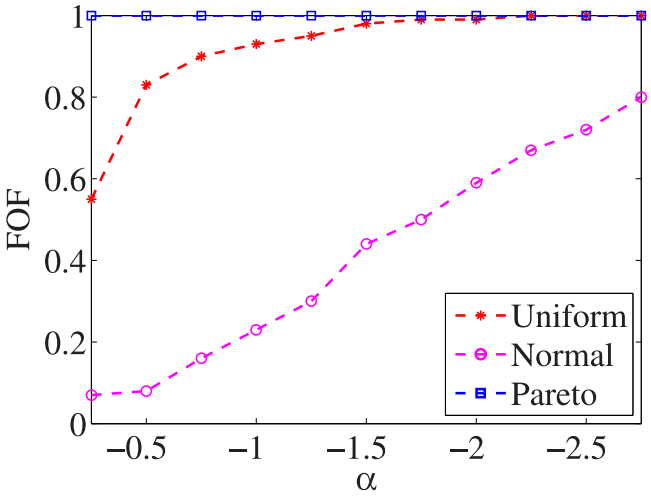

Fig 3. Finding optimal factor of Promoters dataset.

https://doi.org/10.1371/journal.pone.0197564.g003

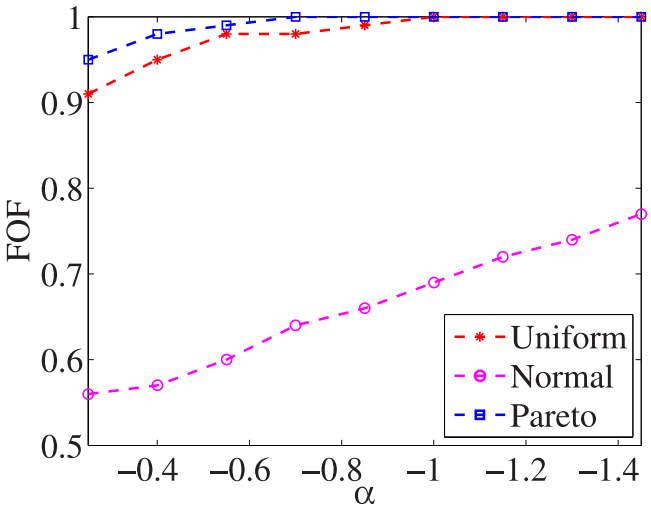

Fig 4. Finding optimal factor of Voting dataset.

https://doi.org/10.1371/journal.pone.0197564.g004

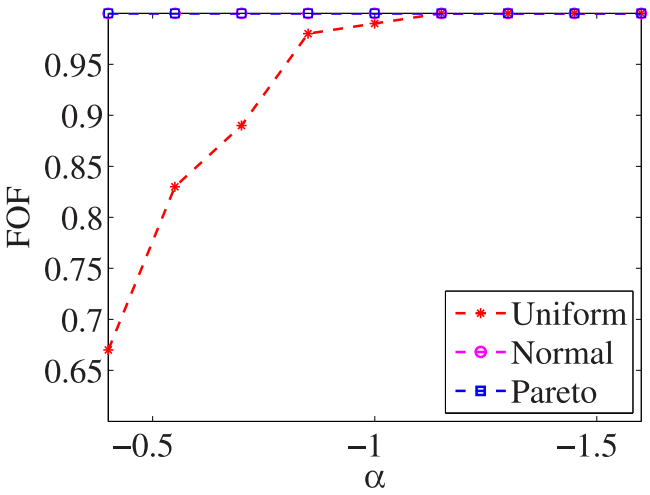

Fig 5. Finding optimal factor of Ionosphere dataset.

https://doi.org/10.1371/journal.pone.0197564.g005

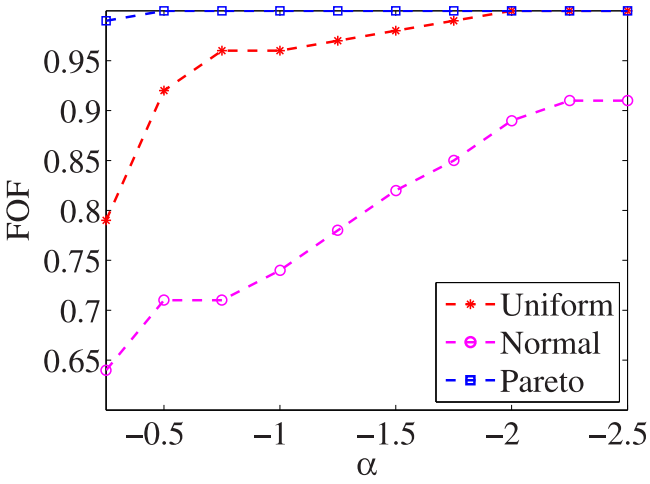

Fig 6. Finding optimal factor of Credit-g dataset.

https://doi.org/10.1371/journal.pone.0197564.g006

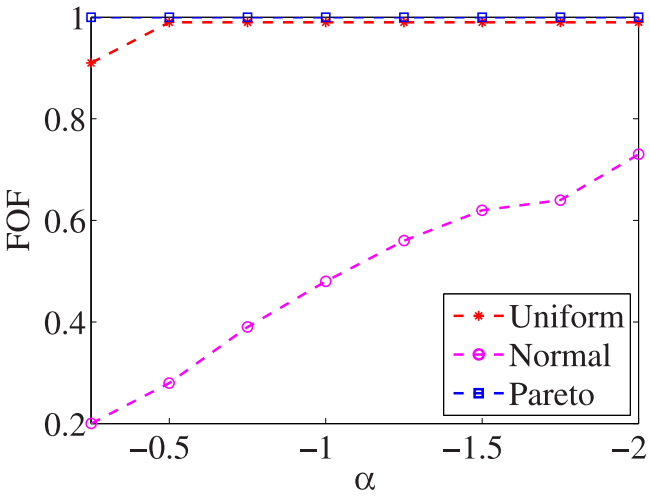

Fig 7. Finding optimal factor of Prostate-GE dataset.

https://doi.org/10.1371/journal.pone.0197564.g007

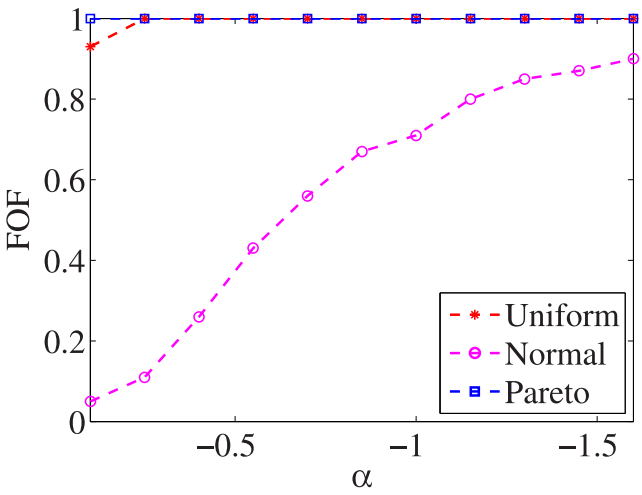

Fig 8. Finding optimal factor of SMK-CAN-187 dataset.

https://doi.org/10.1371/journal.pone.0197564.g008

distribution generates many values close to the mean value, and there are no low test costs and large LSs. Finally, in the uniform distribution, there are more cheap tests than in the normal distribution, and fewer cheap tests than in the pareto distribution.

Figs 10–18 show the average below factor. For the three distributions, the average below factor is more convincing than the maximal below factor because the average below factor is created from a statistical point of view. Hence, for the three distributions, the average below factor can better describe the performance of the CSFS-RSLS algorithm. From these results, we can see that the proposed algorithm obtains the best performance for each dataset from the uniform distribution except for the *SMK-CAN-187* dataset. With the pareto distribution, the average factor is 0 for the *Wpbc*, *Promoters*, *Prostate-GE*, *SMK-CAN-187*, and *Waveform* datasets. These results indicate that the test cost of the feature subset and the optimal reduction is the same. In the *Ionosphere* dataset, although the optimal factor is 1, the average below factor is not 0 but about 0.5. This result shows that the test cost of the feature selection subset is less than the test cost of the optimal reduction and is half that of the optimal reduction.

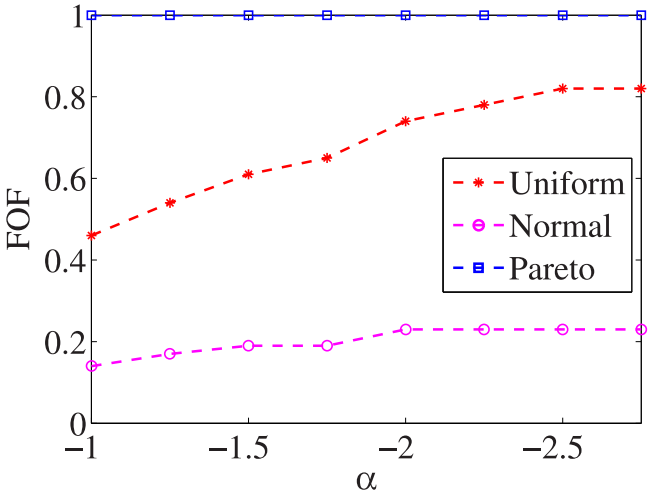

Fig 9. Finding optimal factor of Waveform dataset.

https://doi.org/10.1371/journal.pone.0197564.g009

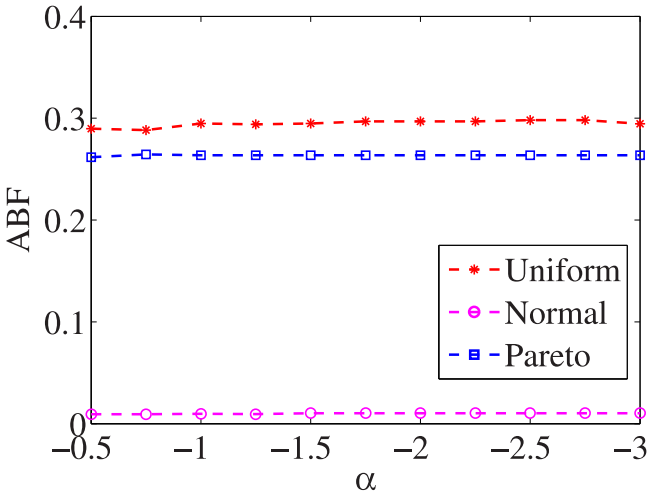

Fig 10. Average below factor of Liver dataset.

https://doi.org/10.1371/journal.pone.0197564.g010

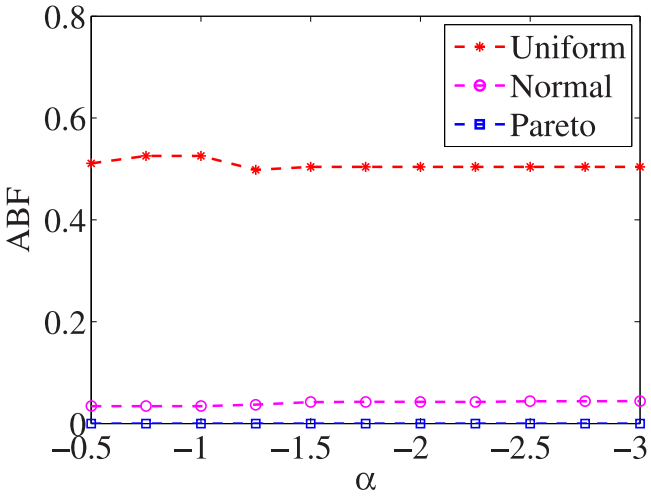

Fig 11. Average below factor of Wpbc dataset.

https://doi.org/10.1371/journal.pone.0197564.g011

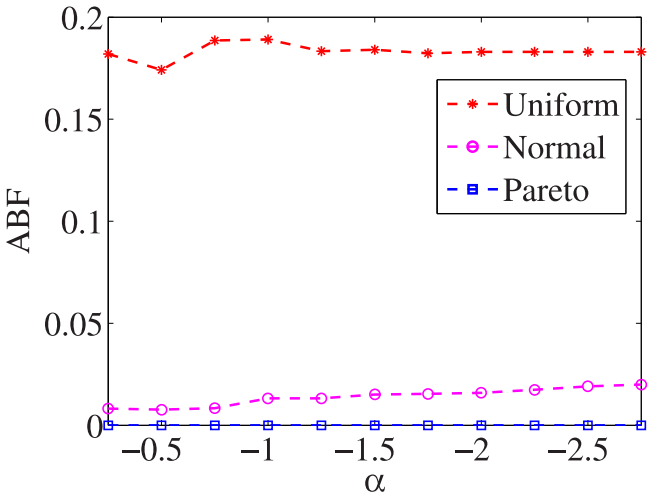

Fig 12. Average below factor of Promoters dataset.

https://doi.org/10.1371/journal.pone.0197564.g012

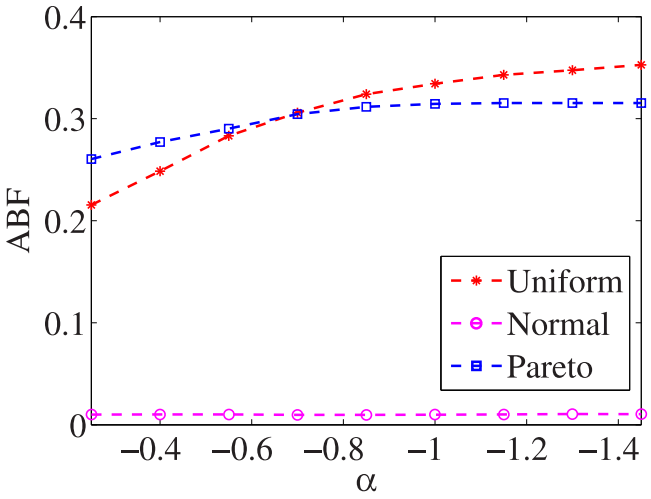

Fig 13. Average below factor of Voting dataset.

https://doi.org/10.1371/journal.pone.0197564.g013

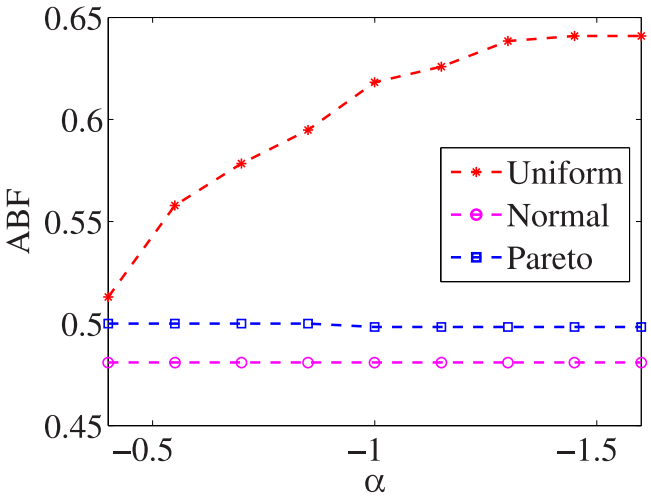

Fig 14. Average below factor of Ionosphere dataset.

https://doi.org/10.1371/journal.pone.0197564.g014

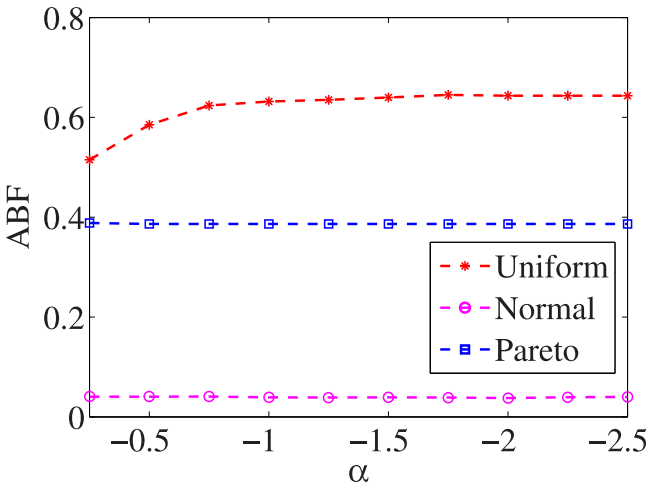

Fig 15. Average below factor of Credit-g dataset.

https://doi.org/10.1371/journal.pone.0197564.g015

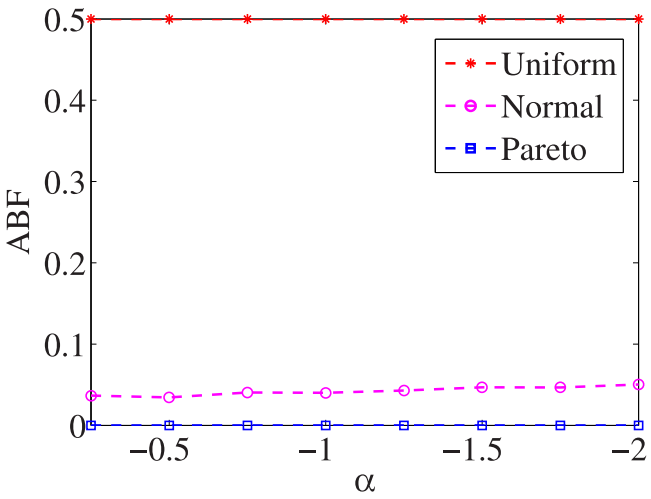

Fig 16. Average below factor of Prostate-GE dataset.

https://doi.org/10.1371/journal.pone.0197564.g016

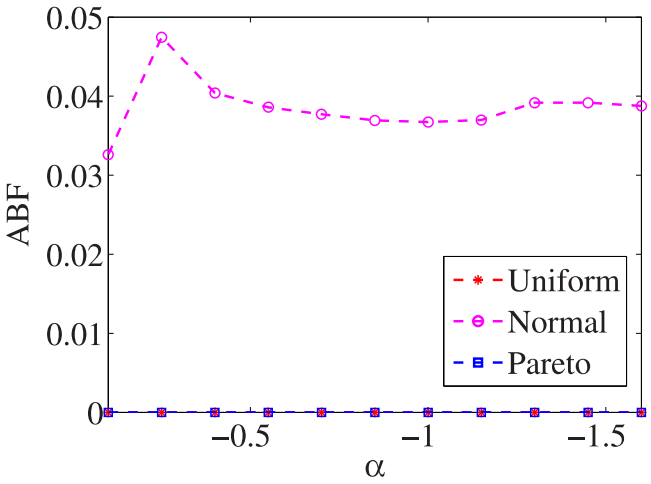

Fig 17. Average below factor of SMK-CAN-187 dataset.

https://doi.org/10.1371/journal.pone.0197564.g017

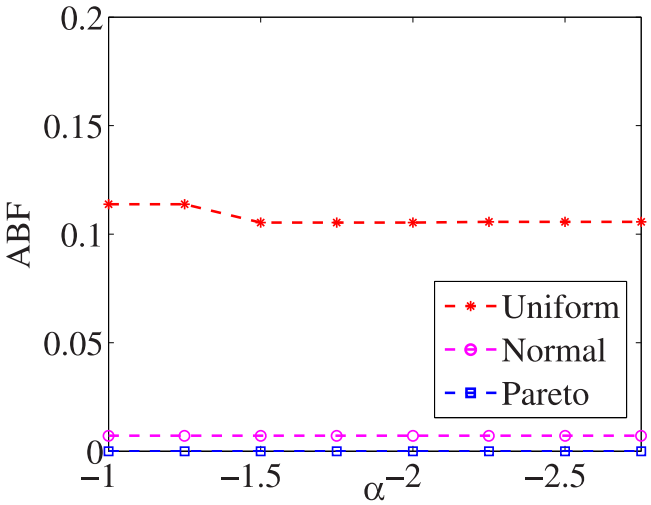

Fig 18. Average below factor of Waveform dataset.

https://doi.org/10.1371/journal.pone.0197564.g018

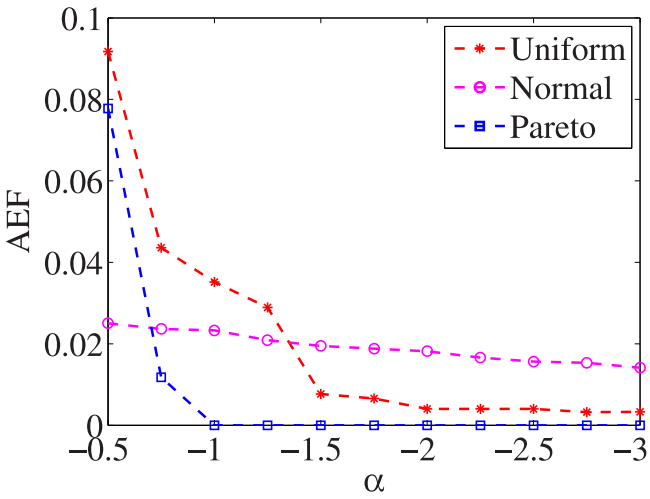

Fig 19. Average exceeding factor of Liver dataset.

https://doi.org/10.1371/journal.pone.0197564.g019

Figs 19–27 show the average exceeding factors. Here, the optimal setting for α is very close, if not equal, to that for the finding optimal factor. We only need to obtain the optimal setting to find the optimal factor. When this α is at the optimal setting, the average exceeding factor is very low. For example, it is 0 for the nine datasets with the pareto distribution. That is, the constructed feature subsets do not have a higher test cost than that of the optimal reduction, on average. This performance would very satisfactory for practical applications.

In Table 7, we list the results of each dataset to compare the two approaches according to the optimal factor. Both methods are based on CSFS-RSLS. The first approach, called the non-weighting approach, is implemented by setting $\alpha = 0$. The second approach, called the average α approach.

We observe the following:

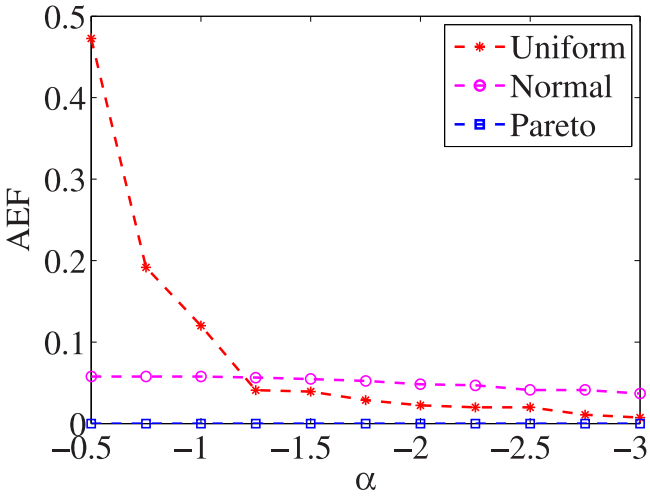

Fig 20. Average exceeding factor of Wpbc dataset.

https://doi.org/10.1371/journal.pone.0197564.g020

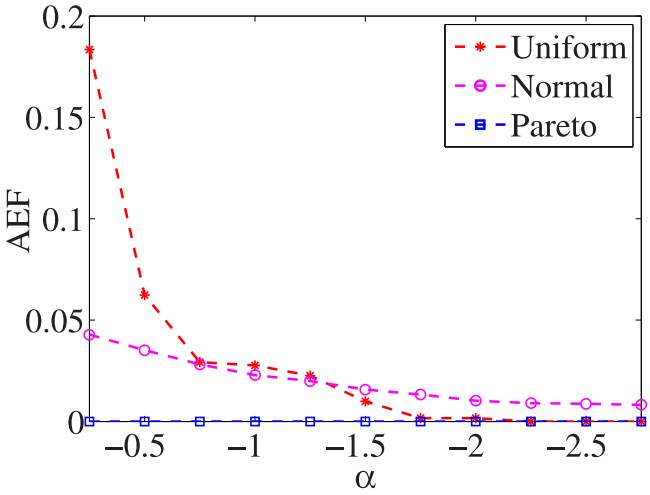

Fig 21. Average exceeding factor of Promoters dataset.

https://doi.org/10.1371/journal.pone.0197564.g021

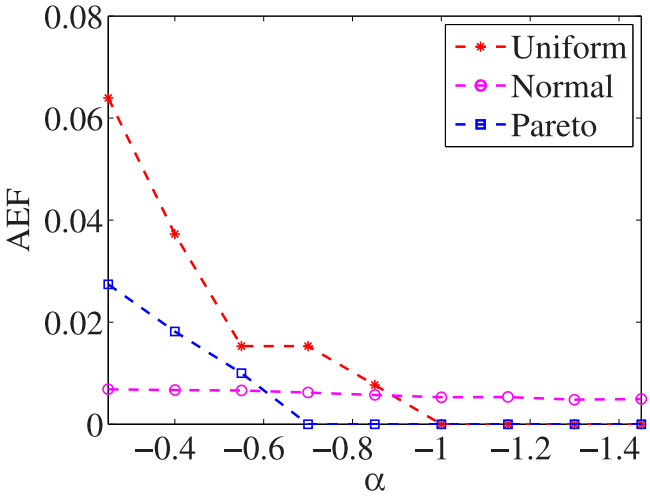

Fig 22. Average exceeding factor of Voting dataset.

https://doi.org/10.1371/journal.pone.0197564.g022

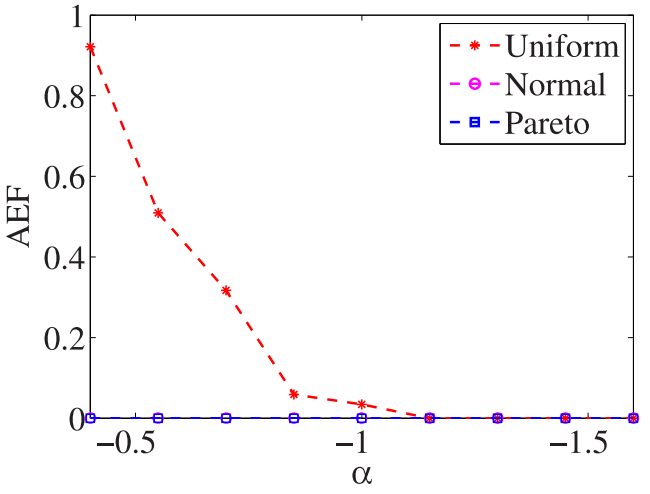

Fig 23. Average exceeding factor of Ionosphere dataset.

https://doi.org/10.1371/journal.pone.0197564.g023

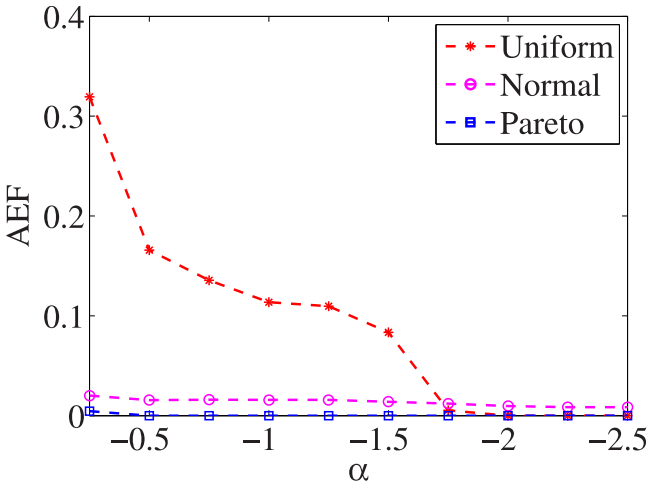

Fig 24. Average exceeding factor of Credit-g dataset.

https://doi.org/10.1371/journal.pone.0197564.g024

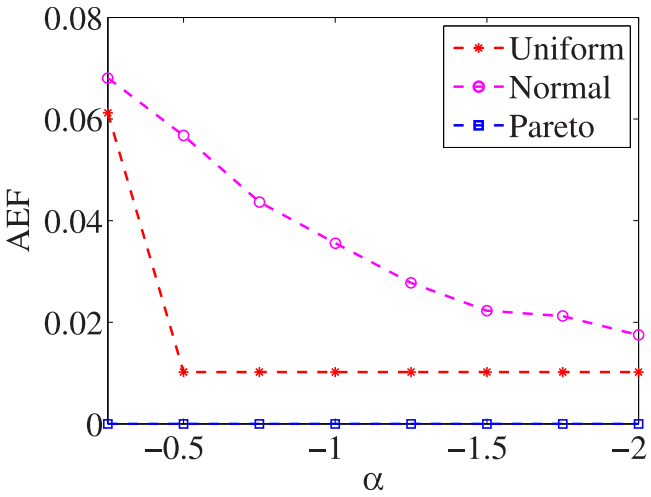

Fig 25. Average exceeding factor of Prostate-GE dataset.

https://doi.org/10.1371/journal.pone.0197564.g025

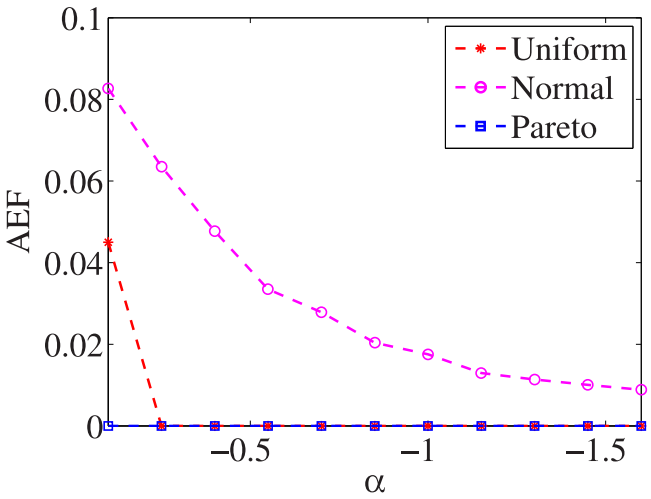

Fig 26. Average exceeding factor of SMK-CAN-187 dataset.

https://doi.org/10.1371/journal.pone.0197564.g026

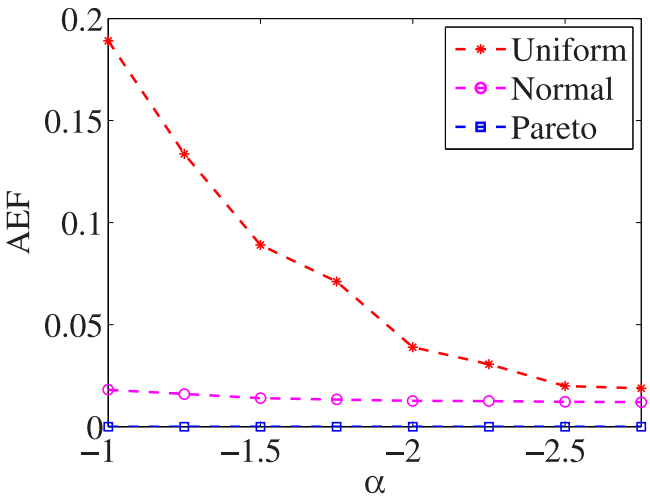

Fig 27. Average exceeding factor of Waveform dataset.

https://doi.org/10.1371/journal.pone.0197564.g027

1. The non-weighting approach only performs well on the *Ionosphere* dataset relative to the other datasets. The non-weighting approach has the highest average value of 0.661 on the *Ionosphere* dataset. For the eight other datasets, the results are unacceptable. For the *Waveform* dataset, when $\alpha = 0$, it obtains an optimal factor of 0 for the uniform and normal distribution. In a word, when $\alpha = 0$, this algorithm has no effect. Therefore, the non-weighting approach is not suitable for the minimal test-cost feature subset problem.
2. The average α approach takes a statistical approach and significantly improves the quality of the results in each dataset. For the *Promoters*, *Prostate-GE*, *Credit-g*, and *SMK-CAN-187* datasets, the approach has especially good results. For the *SMK-CAN-187* dataset, the value increases by about 99.1% for the uniform distribution. Relatively good results are obtained for the other datasets. For example, for the uniform distribution, the best value of the $\alpha = 0$ approach is 0.440, and the value of the average α approach is 0.979, an increase of 52.9%. This result is a big improvement.

Effectiveness compare with two algorithms

In this section, we compare the proposed algorithm with two existing algorithms [43, 44] to show the efficiency of our algorithm. First, the two existing algorithms and the CSFS-RSLS

Table 7. Results for $\alpha = 0$ and α with the optimal setting.

Dataset	$\alpha = 0$			optimal α		
	Uniform	Normal	Pareto	Uniform	Normal	Pareto
Liver	0.145	0.220	0.443	0.894	0.319	0.979
Wpbc	0.018	0.235	0.703	0.854	0.337	1.000
Promoters	0.000	0.040	0.295	0.920	0.415	1.000
Voting	0.440	0.510	0.523	0.979	0.661	0.991
Ionosphere	0.086	1.000	0.898	0.929	1.000	1.000
Credit-g	0.220	0.553	0.548	0.957	0.796	0.999
Prostate-GE	0.003	0.126	0.716	0.980	0.488	1.000
SMK-CAN-187	0.003	0.018	0.741	0.994	0.565	1.000
Waveform	0.000	0.000	0.438	0.678	0.201	1.000

https://doi.org/10.1371/journal.pone.0197564.t007

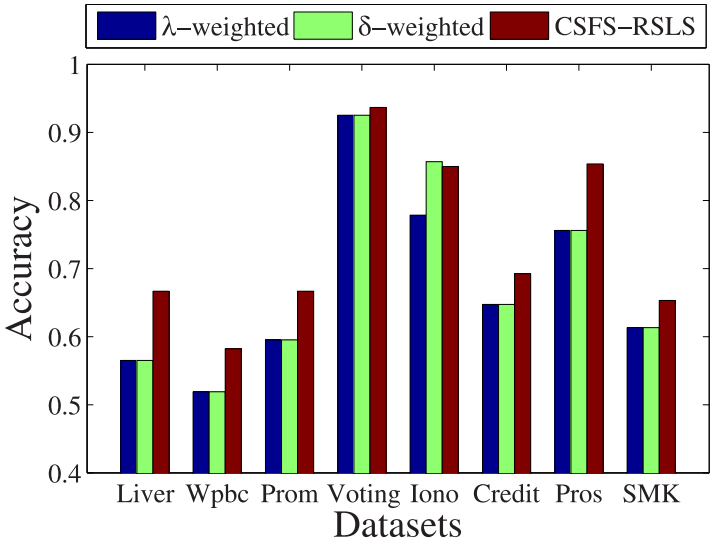

Fig 28. Classification accuracy.

https://doi.org/10.1371/journal.pone.0197564.g028

algorithm are used in a support vector machine classifier to compute the classification accuracy. We used 60% of the dataset as the training set and the rest as the test set. Second, using the uniform distribution, each algorithm was run 100 times with different test cost settings and the optimal factor was compared with different exponential weight settings.

Fig 28 shows the classification accuracy of the three algorithms for eight datasets. For the *Liver*, *Wpbc*, *Promoters*, *Voting*, *Credit*, *Prostate-GE*, and *SMK-CAN-187* datasets, the classification accuracy of the λ-weighted algorithm and δ-weighted algorithm is the same. For these datasets, the CSFS-RSLS algorithms has a higher classification accuracy than these two algorithms. Even for the *Prostate-GE* dataset, the classification accuracy of the CSFS-RSLS algorithm is higher than that of the two algorithms by about 10%. For the *Ionosphere* dataset, our CSFS-RSLS algorithm is only lower than the δ-weighted algorithm by about 1%. However, it is higher than the λ-weighted algorithm by about 7%.

Fig 29 shows the optimal factor found by the three algorithms with the optimal exponential weight. For the *Promoters*, *Voting*, *Ionosphere*, *Credit*, *Prostate-GE*, and *SMK-CAN-187* datasets, the optimal factor found by the CSFS-RSLS algorithm is 1. For the *Ionosphere* dataset, the optimal factor found by the CSFS-RSLS algorithm is higher than that of the λ-weighted algorithm by about 0.4. This is an unsatisfactory number. For the *Liver* dataset, the optimal factor found by the CSFS-RSLS algorithm is lower than that of the δ-weighted algorithm by about 0.01. This value is acceptable.

Conclusion and further work

In this paper, we have developed a new method for cost-sensitive feature selection. Firstly, we use rough sets to calculate the core of all features and use LS to calculate the importance of the each feature. Secondly, the cost is randomly generated by the three different distributions. Finally, we combine the feature importance and cost. To compare the performance of the proposed algorithm, we use two heuristic algorithms to our paper in the same experimental environment. Extensive experimental results show that the proposed algorithm can have better performance and obtain a feature subset with low cost. The CSFS-RSLS algorithm also outperforms the existing algorithms.

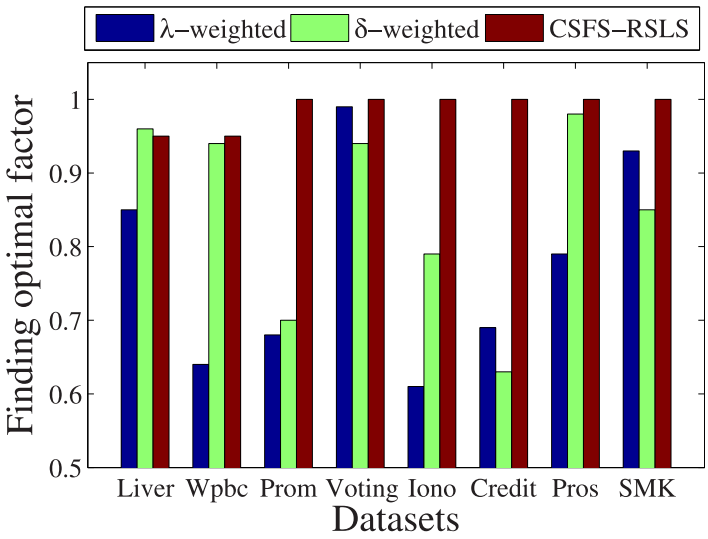

Fig 29. Finding optimal factor.

https://doi.org/10.1371/journal.pone.0197564.g029

With regard to further work, many tasks need to be undertaken. First, other realistic data models with test costs can be built. A second point to be considered in future research is that the misclassification cost also should be added to the model. A model combining the test cost with the misclassification cost will be more suitable for real application. In the future, we will focus on designing more effective and efficient algorithms to cope with the minimal cost feature-selection problem. In summary, this study suggests new research trends for the feature selection problem and cost-sensitive learning.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No.61703196, and the Natural Science Foundation of Fujian Province under Grant No.2018J01549. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Author Contributions

Conceptualization: Hong Zhao.

Writing – original draft: Shenglong Yu.

Writing – review & editing: Hong Zhao.

References

1. Dai JH, Hu QH, Zhang JH, Hu H, Zheng NG. Attribute selection for partially labeled categorical data by rough set approach. IEEE Transactions on Cybernetics. 2017;PP(99):1–12.
2. Rückstieß T, Osendorfer C, van der Smagt P. Minimizing data consumption with sequential online feature selection. International Journal of Machine Learning and Cybernetics. 2013; 4(3):235–243. https://doi.org/10.1007/s13042-012-0092-x
3. Subrahmanya N, Shin YC. A variational bayesian framework for group feature selection. International Journal of Machine Learning and Cybernetics. 2013; 4(6):609–619. https://doi.org/10.1007/s13042-012-0121-9

4. Xie Z, Xu Y. Sparse group LASSO based uncertain feature selection. International Journal of Machine Learning and Cybernetics. 2014; 5(2):201–210. https://doi.org/10.1007/s13042-013-0156-6
5. Golberg DE. Genetic algorithms in search, optimization, and machine learning. Addion Wesley. 1989; 102.
6. Li J, Mei C, Lv Y. Incomplete decision contexts: approximate concept construction, rule acquisition and knowledge reduction. International Journal of Approximate Reasoning. 2013; 54(1):149–165. https://doi.org/10.1016/j.ijar.2012.07.005
7. Zhu P, Zhu W, Wang W, Zuo W, Hu Q. Non-convex regularized self-representation for unsupervised feature selection. Image and Vision Computing. 2016; 60:22–29. https://doi.org/10.1016/j.imavis.2016.11.014
8. Alhaj TA, Siraj MM, Zainal A, Elshoush HT, Elhaj F. Feature selection using information gain for improved structural-based alert correlation. PloS one. 2016; 11(11):e0166017. https://doi.org/10.1371/journal.pone.0166017 PMID: 27893821
9. Hu Q, Liu J, Yu D. Mixed feature selection based on granulation and approximation. Knowledge-Based Systems. 2008; 21(4):294–304. https://doi.org/10.1016/j.knosys.2007.07.001
10. Zhong N, Dong J, Ohsuga S. Using rough sets with heuristics for feature selection. Journal of Intelligent Information Systems. 2001; 16(3):199–214. https://doi.org/10.1023/A:1011219601502
11. Lin C, Chen W, Qiu C, Wu Y, Krishnan S, Zou Q. LibD3C: ensemble classifiers with a clustering and dynamic selection strategy. Neurocomputing. 2014; 123:424–435. https://doi.org/10.1016/j.neucom.2013.08.004
12. Liu H, Motoda H. Feature selection for knowledge discovery and data mining. vol. 454. Springer Science & Business Media; 2012.
13. Martina F, Beccuti M, Balbo G, Cordero F. Peculiar Genes Selection: a new features selection method to improve classification performances in imbalanced data sets. PloS one. 2017; 12(8):e0177475. https://doi.org/10.1371/journal.pone.0177475 PMID: 28806759
14. Wan S, Duan Y, Zou Q. HPSLPred: an ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source. Proteomics. 2017; 17:1–12. https://doi.org/10.1002/pmic.201700262
15. Weiss Y, Elovici Y, Rokach L. The cash algorithm-cost-sensitive attribute selection using histograms. Information Sciences. 2013; 222:247–268. https://doi.org/10.1016/j.ins.2011.01.035
16. Zou Q, Zeng J, Cao L, Ji R. A novel features ranking metric with application to scalable visual and bioinformatics data classification. Neurocomputing. 2016; 173:346–354. https://doi.org/10.1016/j.neucom.2014.12.123
17. Cortes C, Vapnik V. Support-vector networks. Machine Learning. 1995; 20(3):273–297. https://doi.org/10.1007/BF00994018
18. Bazan JG, Skowron A, Synak P. Dynamic reducts as a tool for extracting laws from decisions tables. In: International Symposium on Methodologies for Intelligent Systems. vol. 869. Springer; 1994. p. 346–355.
19. Liu M, Zhang D. Feature selection with effective distance. Neurocomputing. 2016; 215:100–109. https://doi.org/10.1016/j.neucom.2015.07.155
20. Maji P, Paul S. Rough set based maximum relevance-maximum significance criterion and gene selection from microarray data. International Journal of Approximate Reasoning. 2011; 52(3):408–426. https://doi.org/10.1016/j.ijar.2010.09.006
21. Wang S, Pedrycz W, Zhu Q, Zhu W. Subspace learning for unsupervised feature selection via matrix factorization. Pattern Recognition. 2015; 48(1):10–19. https://doi.org/10.1016/j.patcog.2014.08.004
22. Zhu P, Zhu W, Hu Q, Zhang C, Zuo W. Subspace clustering guided unsupervised feature selection. Pattern Recognition. 2017; 66:364–374. https://doi.org/10.1016/j.patcog.2017.01.016
23. Fumera G, Roli F. Cost-sensitive learning in support vector machines. Convegno Associazione Italiana per Intelligenza Artificiale. 2002.
24. Ling CX, Yang Q, Wang J, Zhang S. Decision trees with minimal costs. In: Proceedings of the Twenty-First International Conference on Machine Learning. ACM; 2004. p. 69.
25. Wan J, Yang M, Chen Y. Discriminative cost sensitive laplacian score for face recognition. Neurocomputing. 2015; 152:333–344. https://doi.org/10.1016/j.neucom.2014.10.059
26. Turney PD. Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm. Journal of Artificial Intelligence Research. 1995; 2:369–409.
27. Emary E, Zawbaa HM, Hassanien AE. Binary ant lion approaches for feature selection. Neurocomputing. 2016; 213:54–65. https://doi.org/10.1016/j.neucom.2016.03.101

28. Greiner R, Grove AJ, Roth D. Learning cost-sensitive active classifiers. Artificial Intelligence. 2002; 139(2):137–174. https://doi.org/10.1016/S0004-3702(02)00209-6
29. Ji S, Carin L. Cost-sensitive feature acquisition and classification. Pattern Recognition. 2007; 40 (5):1474–1485. https://doi.org/10.1016/j.patcog.2006.11.008
30. He H, Min F, Zhu W. Attribute reduction in test-cost-sensitive decision systems with common-test-costs. In: Proceedings of the 3rd International Conference on Machine Learning and Computing. vol. 1; 2011. p. 432–436.
31. Min F, He H, Qian Y, Zhu W. Test-cost-sensitive attribute reduction. Information Sciences. 2011; 181 (22):4928–4942. https://doi.org/10.1016/j.ins.2011.07.010
32. Susmaga R. Computation of minimal cost reducts. In: International Symposium on Methodologies for Intelligent Systems. Springer; 1999. p. 448–456.
33. He H, Min F. Accumulated cost based test-cost-sensitive attribute reduction. In: International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing. Springer; 2011. p. 244–247.
34. Pan G, Min F, Zhu W. A genetic algorithm to the minimal test cost reduct problem. In: 2011 IEEE International Conference on Granular Computing; 2011. p. 539–544.
35. Tan M. Cost-sensitive learning of classification knowledge and its applications in robotics. Machine Learning. 1993; 13(1):7–33. https://doi.org/10.1007/BF00993101
36. Zhang S. Cost-sensitive classification with respect to waiting cost. Knowledge-Based Systems. 2010; 23(5):369–378. https://doi.org/10.1016/j.knosys.2010.01.008
37. Jia X, Liao W, Tang Z, Shang L. Minimum cost attribute reduction in decision-theoretic rough set models. Information Sciences. 2013; 219:151–167. https://doi.org/10.1016/j.ins.2012.07.010
38. Zhao H, Min F, Zhu W. A backtracking approach to minimal cost feature selection of numerical data. Journal of Information & Computational Science. 2013; 10:4105–4115.
39. Yang X, Qi Y, Song X, Yang J. Test cost sensitive multigranulation rough set: model and minimal cost selection. Information Sciences. 2013; 250:184–199. https://doi.org/10.1016/j.ins.2013.06.057
40. Min F, Xu J. Semi-greedy heuristics for feature selection with test cost constraints. Granular Computing. 2016; p. 1–13.
41. Johnson RA, Wichern DW. Applied multivariate statistical analysis. Pearson Education Limited Essex; 2014.
42. Min F, Zhu W, Zhao H, Pan G, Liu J, Xu Z. Coser: cost-senstive rough sets; 2016.
43. Min F, Zhu W. Attribute reduction of data with error ranges and test costs. Information Sciences. 2012; 211:48–67. https://doi.org/10.1016/j.ins.2012.04.031
44. Zhao H, Min F, Zhu W. Test-cost-sensitive attribute reduction of data with normal distribution measurement errors. Mathematical Problems in Engineering. 2013; 2013:1–13.
45. Min F, Liu QH. A hierarchical model for test-cost-sensitive decision systems. Information Sciences. 2009; 179:2442–2452. https://doi.org/10.1016/j.ins.2009.03.007
46. Zhu W. Relationship between generalized rough sets based on binary relation and covering. Information Sciences. 2009; 179(3):210–225. https://doi.org/10.1016/j.ins.2008.09.015
47. Qian Y, Liang J, Pedrycz W, Dang C. Positive approximation: an accelerator for attribute reduction in rough set theory. Artificial Intelligence. 2010; 174(9-10):597–618. https://doi.org/10.1016/j.artint.2010.04.018
48. Deng D. Parallel reduct and its properties. In: 2009, GRC'09. IEEE International Conference on Granular Computing. IEEE; 2009. p. 121–125.
49. Deng D, Wang J, Li X. Parallel reducts in a series of decision subsystems. In: 2009. CSO 2009. International Joint Conference on Computational Sciences and Optimization. vol. 2. IEEE; 2009. p. 377–380.
50. Zhao Y, Luo F, Wong SM, Yao Y. A general definition of an attribute reduct. In: International Conference on Rough Sets and Knowledge Technology. Springer; 2007. p. 101–108.
51. He X, Cai D, Niyogi P. Laplacian score for feature selection. In: Advances in Neural Information Processing Systems. vol. 18; 2005. p. 507–514.
52. Spira A, Beane JE, Shah V, Steiling K, Liu G, Schembri F, et al. Airway epithelial gene expression in the diagnostic evaluation of smokers with suspect lung cancer. Nature Medicine. 2007; 13(3):361–366. https://doi.org/10.1038/nm1556 PMID: 17334370
53. Blake C, Merz CJ. {UCI} repository of machine learning databases. 1998.