Association for Information Systems
AIS Electronic Library (AISeL)

ICEB 2018 Proceedings

International Conference on Electronic Business (ICEB)

Winter 12-6-2018

Unsupervised Feature Selection Algorithm via Local Structure Learning and Kernel Function

Jiaye Li
Guangxi Normal University, China, jiaye_ligxnu@126.com

Xiaofeng Zhu
Guangxi Normal University, China, seanzhuxf@gmail.com

Jiangzhang Gan
Guangxi Normal University, China, ganjz@outlook.com

Leyuan Zhang
Guangxi Normal University, China, 846390062@qq.com

Shanwen Zhang
Guangxi Normal University, China, 244491777@qq.com

See next page for additional authors

Authors
Jiaye Li, Xiaofeng Zhu, Jiangzhang Gan, Leyuan Zhang, Shanwen Zhang, and Shichao Zhang

Li, J.Y., Zhu, X.F., Gan, J.Z., Zhang, L.Y., Zhang, S.W. & Zhang, S.C. (2018). Unsupervised feature selection algorithm via local structure learning and kernel function. In *Proceedings of The 18th International Conference on Electronic Business* (pp. 721-730). ICEB, Guilin, China, December 2-6.

Unsupervised Feature Selection Algorithm via Local Structure Learning and Kernel Function

(Full Paper)

Jiaye. Li, Guangxi Normal University, China, jiaye_ligxnu@126.com
Xiaofeng. Zhu*, Guangxi Normal University, China, seanzhuxf@gmail.com
Jiangzhang. Gan, Guangxi Normal University, China, ganjz@outlook.com
Leyuan. Zhang, Guangxi Normal University, China, 846390062@qq.com
Shanwen. Zhang, Guangxi Normal University, China, 244491777@qq.com
Shichao. Zhang*, Guangxi Normal University, China, zhangsc@mailbox.gxnu.edu.cn

ABSTRACT

In order to reduce dimensionality of high-dimensional data, a series of feature selection algorithms have been proposed. But these algorithms have the following disadvantages: (1) they do not fully consider the nonlinear relationship between data features (2) they do not consider the similarity between data features. To solve the above two problems, we propose an unsupervised feature selection algorithm based on local structure learning and kernel function. First, through the kernel function, we map each feature of the data to the kernel space, so that the nonlinear relationship of the data features can be fully exploited. Secondly, we apply the theory of local structure learning to the features of data, so that the similarity of data features is considered. Then we added a low rank constraint to consider the global information of the data. Finally, we add sparse learning to make feature selection. The experimental results show that the proposed algorithm has better results than the comparison methods.

Keywords: Feature selection · Kernel function · Sparse learning · Local structure learning

*Corresponding author

INTRODUCTION

With the development of computer science and technology, the information age is coming. At the same time, a large number of high-dimensional data are brought (Zhu *et al.*, 2014). Artificial intelligence, data mining and other fields are also booming (Zhang *et al.*, 2018). It is very difficult for people to deal with thousands of data, sometimes it will bring problems of dimensional disaster (Zhu *et al.*, 2013; Zhu *et al.*, 2010). For these high-dimensional data, people must preprocess it. The feature selection is one of the most effective ways (Bolón-Canedo *et al.*, 2016). It is necessary to preprocess the data through feature selection to narrow the data dimension (Ling *et al.*, 2004).

Feature selection (Zhu *et al.*, 2013) includes linear feature selection and nonlinear feature selection (Zhang *et al.*, 2007). Their fundamental purpose is to find a relatively small and representative subset of features (Qin *et al.*, 2007; Sheeja *et al.*, 2018). There are many commonly used feature selection methods (Zhang *et al.*, 2011), but they cannot unearth the nonlinear relationship between data features. The local structure learning is applied to the sample at the beginning, and the structure between the samples is fully embodied by constructing the similarity matrix between the samples (Nie *et al.*, 2016), so as to achieve better experimental results. But it does not fully embody the structural relationship between features. To this end, we maps each feature of the data to a high-dimensional space by a kernel function, so that the nonlinear relationship between them is linearly separable in the high-dimensional space. At the same time, the local structure learning is applied to the data features. To better represent the local structural relationship of data features in low-dimensional space. A more efficient feature selection algorithm is proposed, which is called Unsupervised Feature Selection Algorithm via Local Structure Learning and Kernel Function (LSK FS).

This paper firstly processes the data through the kernel function to obtain the kernel matrix, which solves the limitation that the linear feature selection can be only performed. Secondly, it constructs the similarity matrix for the data feature to perform the local structure learning, it can improve the classification accuracy. Low rank constraints can eliminate noise interference. Finally we use l_1 – norm of a vector for feature selection. Because this paper considers the nonlinear relationship and similarity between data features at the same time, it has better effect than the single linear feature selection method. The experimental results show that the algorithm can achieve better results in classification accuracy.

The algorithm proposed in this paper has the following advantages:

(1) Since the general feature selection algorithm can only find the linear relationship between data features, it cannot find the nonlinear relationship between data features. Therefore, the algorithm maps each feature of the data matrix to a kernel matrix through the kernel function, so as to fully exploit the complex nonlinear relationship between the data features in the kernel space. Furthermore, the relationship between data features is more thoroughly explored.

(2) Different from ordinary local structure learning, it only calculates the optimal result of the similarity relationship between samples. Our algorithm is aimed at data features, and the similar matrix learning and low-dimensional space learning are alternated to achieve the optimal feature selection effect.

(3) Low rank constraints can significantly reduce the amount of computation, while low rank characterizes the degree of data redundancy. Noise samples increase the rank of the coefficient matrix, and low rank constraints can reduce noise interference, while low rank is a consideration of the global structure of the data. It can improve the operating efficiency and classification accuracy of the algorithm.

OUR METHOD

In this section, we first introduce the symbols used in this article and then explain our proposed LSK FS algorithm, in Sections 2.1 and 2.2, respectively, and then elaborate the proposed optimization method in Section 2.3.

2.1 Notations

For the data matrix $\boldsymbol{X} \in \boldsymbol{R}^{n \times d}$, the *i*-th row and the *j*-th column are denoted as $\boldsymbol{X}^i$ and $\boldsymbol{X}_j$ respectively, and the elements of the *i*-th row and the *j*-th column are denoted as $x_{i,j}$. The trace of the matrix $\boldsymbol{X}$ is denoted by $tr(\boldsymbol{X})$, $\boldsymbol{X}^T$ denotes the transpose of the matrix $\boldsymbol{X}$, and $\boldsymbol{X}^{-1}$ represents the inverse of the matrix $\boldsymbol{X}$.

2.2 LSK FS Algorithm

Suppose a given sample data set $\boldsymbol{X} \in \boldsymbol{R}^{n \times d}$, where *n* and *d* represent the number of samples and the number of attributes, respectively. This paper first breaks the data set $\boldsymbol{X} \in \boldsymbol{R}^{n \times d}$ into *d* column vectors, each vector $x_i \in R^{n \times 1}, i = 1,...,d$. Then it treats each element in each x_i as an independent feature value $x_{ij} \in R, j = 1,...,n$. And projects them into the kernel space to get the kernel matrix $\boldsymbol{K}^{(i)} \in \boldsymbol{R}^{n \times n}$, namely:

$$\boldsymbol{K}^{(i)} = \begin{bmatrix} k(x_{i1}, x_{i1}) & k(x_{i1}, x_{i2}) & \dots & k(x_{i1}, x_{in}) \\ k(x_{i2}, x_{i1}) & k(x_{i2}, x_{i2}) & \dots & k(x_{i2}, x_{in}) \\ \dots & \dots & \dots & \dots \\ k(x_{in}, x_{i1}) & k(x_{in}, x_{i2}) & \dots & k(x_{in}, x_{in}) \end{bmatrix} \tag{1}$$

Thus the original $\boldsymbol{X} \in \boldsymbol{R}^{n \times d}$ becomes *d* kernel matrix.

The unsupervised feature selection algorithm mainly mines more representative features in the data. In the absence of the class label $\boldsymbol{Y}$, using the data matrix $\boldsymbol{X}$ as a response matrix, the internal structure of the original features of the data can be better preserved (Almusallam *et al.*, 2018; Xue *et al.*, 2016). In order to fully exploit the nonlinear relationship of data features. Get the following expression:

$$\boldsymbol{X} = \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \tag{2}$$

Where: $\boldsymbol{W} \in \boldsymbol{R}^{n \times d}$ represents the kernel coefficient matrix; $\boldsymbol{\alpha} \in \boldsymbol{R}^{d \times 1}$ is used to perform feature selection, which is equivalent to the weight vector of the feature; α_i is an element of the vector $\boldsymbol{\alpha}$; $\boldsymbol{K}^{(i)} \in \boldsymbol{R}^{n \times n}$ is the kernel matrix.

Predecessors have proved that the local structure between data can be used to reduce the dimension (Liu *et al.*, 2017), so this paper makes local structure learning by establishing a similarity matrix between data features in low-dimensional space. The following formula is obtained through local structure learning:

$$\min_{\boldsymbol{z}} \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} \tag{3}$$

Where $x_i \in R^{n \times 1}$ represents *i*-th feature. $\boldsymbol{W} \in \boldsymbol{R}^{n \times d}$ is the conversion matrix of high-dimensional data in low-dimensional space, $s_{i,j}$ is an element of matrix $\boldsymbol{S}$, indicating the similarity between feature x_i and feature x_j. If the feature x_i is the *k*-th nearest neighbor of the feature x_j, then the value $s_{i,j}$ is obtained by the Gaussian kernel function; otherwise =0. In order to make $\boldsymbol{X}$ get a better fitting effect, and consider the structural relationship between data features in low-dimensional space, we get the following formula:

$$\min_{\boldsymbol{S},\boldsymbol{W},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} \tag{4}$$

Since the similarity matrix $\boldsymbol{S}$ is particularly affected by the influence of parameters σ. In order to reduce the number of adjustment parameters, a more efficient similarity matrix is learned. In this paper, structural learning and low-dimensional space learning are alternated to achieve their optimal results. Specifically get the following formula:

$$\begin{aligned} &\min_{\boldsymbol{S},\boldsymbol{W},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{W} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{W} - x_j^T \boldsymbol{W} \right\|_2^2 s_{i,j} + \lambda_2 \left\| s_i \right\|_2^2 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, \mathrm{s}_{i,i} = 0, s_{i,j} \geq 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0 \end{aligned} \tag{5}$$

Among them, λ_1, λ_2 is the tuning parameter, s_i is the i-th column of the similar matrix $\boldsymbol{S}$, and $\|s_i\|_2^2$ is used to avoid unimportant results. $\mathbf{1}$ represents a vector with all elements of 1. $\mathrm{N}(i)$ represents a set of neighbors the i-th feature. In order to maintain rotation invariance, we set $s_i^T \mathbf{1} = 1$. Therefore, the above formula can make the value $s_{i,j}$ corresponding to the feature closer to the distance larger, and the value $s_{i,j}$ corresponding to the feature whose distance is farther is smaller.

In order to eliminate the interference of the outliers, the noise samples are removed at the same time (Wan *et al.*, 2018). This paper adds a low rank constraint to the matrix $\boldsymbol{W}$ (Li *et al.*, 2017), namely:

$$\boldsymbol{W} = \boldsymbol{AB} \tag{6}$$

Among them, $\boldsymbol{A} \in \boldsymbol{R}^{m \times r}$, $\boldsymbol{B} \in \boldsymbol{R}^{r \times d}$, $r \leq \min(n, d)$, we also orthogonally limit the matrix $\boldsymbol{A}$, in order to fully consider the correlation between the output variables. We add a l_1 – norm of $\boldsymbol{\alpha}$ for sparse learning and feature selection (Tsagris *et al.*, 2018). Finally we get our final objective function as follows:

$$\begin{aligned} &\min_{\boldsymbol{S},\boldsymbol{A},\boldsymbol{B},\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &+ \lambda_2 \left\| s_i \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, \mathrm{s}_{i,i} = 0, \\ &s_{i,j} \geq 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0, \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{7}$$

Among them, $\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \in \boldsymbol{R}^{r \times r}$, λ_1, λ_2 and λ_3 are the tuning parameter. The kernel matrix $\boldsymbol{K}$ is calculated by the Gaussian kernel function, and its main function is to map the data to the kernel space, thereby mining the nonlinear relationship between the data features. The l_1 – norm of the last item $\boldsymbol{\alpha}$ is used to sparse the features for feature selection. If the value of the element corresponding to the vector $\boldsymbol{\alpha}$ is zero, it means that the feature is not selected.

2.3 Optimization

Since the objective function is not co-convex, the closed solution cannot be directly obtained. Therefore, this paper proposes an alternate iterative optimization method to solve the problem, which is divided into the following four steps:

Update $\boldsymbol{A}$ by fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{B}$:

When $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{B}$ are fixed, the optimization (7) problem becomes:

$$\begin{aligned} &\min_{\boldsymbol{A}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{AB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{8}$$

We make $\boldsymbol{P} = \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)}$, then the (8) formula can be transformed into:

$$\begin{aligned} &\min_{\boldsymbol{A}} \left\| \boldsymbol{X} - \boldsymbol{PAB} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{AB} - x_j^T \boldsymbol{AB} \right\|_2^2 s_{i,j} \\ &s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{9}$$

We simplify the (9), we have:

$$\begin{aligned} &\min_{\boldsymbol{A}} tr(\boldsymbol{X}^T \boldsymbol{X} - \boldsymbol{X}^T \boldsymbol{PAB} - \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{X} + \boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{P}^T \boldsymbol{PAB}) \\ &+ \lambda_1 tr(\boldsymbol{B}^T \boldsymbol{A}^T \boldsymbol{XLX}^T \boldsymbol{AB}), s.t., \boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I} \end{aligned} \tag{10}$$

Where $\mathrm{tr}(\cdot)$ represents the trace of matrix, $\boldsymbol{L} = \boldsymbol{Q} - \boldsymbol{S} \in \boldsymbol{R}^{d \times d}$ is a Laplace matrix, $\boldsymbol{Q}$ is a diagonal matrix, and the elements of each column are $q_{i,i} = \sum_{j=1}^{d} s_{i,j}$. Deriving for $\boldsymbol{A}$, we have:

$$2\lambda_1 \boldsymbol{XLX}^T \boldsymbol{ABB}^T - 2\boldsymbol{P}^T \boldsymbol{XB}^T + 2\boldsymbol{P}^T \boldsymbol{PABB}^T \tag{11}$$

Due to the orthogonality of $\boldsymbol{A}$, we can optimize it by the method in (Zhao H *et al.*, 2016).

Update $\boldsymbol{B}$ by fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{A}$

By fixing $\boldsymbol{S}$, $\boldsymbol{\alpha}$ and $\boldsymbol{A}$, the objective function (7) can be simplified as follows:

$$\min_{\boldsymbol{B}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{A}\boldsymbol{B} \right\|_F^2 + \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{A}\boldsymbol{B} - x_j^T \boldsymbol{A}\boldsymbol{B} \right\|_2^2 s_{i,j} \tag{12}$$

It is easy to get (12) is equivalent to the following formula:

$$\begin{aligned} &\min_{\boldsymbol{B}} tr(\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{X}^T\boldsymbol{P}\boldsymbol{A}\boldsymbol{B} - \boldsymbol{B}^T\boldsymbol{A}^T\boldsymbol{P}^T\boldsymbol{X} + \boldsymbol{B}^T\boldsymbol{A}^T\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{A}\boldsymbol{B}) \\ &+\lambda_1 tr(\boldsymbol{B}^T\boldsymbol{A}^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{B}) \end{aligned} \tag{13}$$

When we ask for $\boldsymbol{B}$ and let its derivative be zero, we can get:

$$\boldsymbol{B} = (\boldsymbol{A}^T\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{A} + \lambda_1 \boldsymbol{A}^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{P}^T\boldsymbol{X} \tag{14}$$

Update S by fixing A, α and B

After fixing $\boldsymbol{A}$, $\boldsymbol{\alpha}$ and $\boldsymbol{B}$, the objective function (7) becomes:

$$\begin{aligned} &\min_{\boldsymbol{S}} \lambda_1 \sum_{i,j}^{d} \left\| x_i^T \boldsymbol{A}\boldsymbol{B} - x_j^T \boldsymbol{A}\boldsymbol{B} \right\|_2^2 s_{i,j} + \lambda_2 \left\| s_i \right\|_2^2 \\ &s.t., \forall i, s_i^T \mathbf{1} = 1, \mathrm{s}_{i,i} = 0, \\ &s_{i,j} \geq 0, \text{if } j \in \mathrm{N}(i), otherwise\ 0 \end{aligned} \tag{15}$$

We first calculate the Euclidean distance between every two data features to construct the neighbors of all the features. If the j-th feature does not belong to the nearest neighbor of the i-th feature, then the value of $s_{i,j}$ is zero; otherwise, the value of $s_{i,j}$ is solved by equation (18).

At the same time, optimizing $\boldsymbol{S}$ is equivalent to optimizing each $s_i (i = 1, ..., d)$ individually, so we further translate the optimization problem into the following equation:

$$\min_{s_i^T 1 = 1, s_{i,i} = 0, s_{i,j} \geq 0} \sum_{i,j}^{d} (\lambda_1 \left\| x_i^T \boldsymbol{A}\boldsymbol{B} - x_j^T \boldsymbol{A}\boldsymbol{B} \right\|_2^2 s_{i,j} + \lambda_2 s_{i,j}^2) \tag{16}$$

Here, $\boldsymbol{Z} \in \boldsymbol{R}^{d \times d}$, in which $\boldsymbol{Z}_{i,j} = \lambda_1 \left\| x_i^T \boldsymbol{A}\boldsymbol{B} - x_j^T \boldsymbol{A}\boldsymbol{B} \right\|_2^2$, such (16) further becomes:

$$\min_{s_i^T 1 = 1, s_{i,i} = 0, s_{i,j} \geq 0} \left\| s_i + \frac{\lambda_1}{2\lambda_2} \boldsymbol{Z}_i \right\|_2^2 \tag{17}$$

Under KKT conditions, we can get the following:

$$s_{i,j} = (-\frac{\lambda_1}{2\lambda_2} Z_{i,j} + \tau)_+ \tag{18}$$

Since each data feature has a neighbor, we sort each $Z_i (i = 1, ...d)$ in descending order, that is $\hat{Z}_i = \{\hat{Z}_{i,1}, ..., \hat{Z}_{i,d}\}$, we know: $s_{i,k+1} = 0$, $s_{i,k} > 0$. We have:

$$-\frac{\lambda_1}{2\lambda_2} \hat{Z}_{i,k+1} + \tau \leq 0 \tag{19}$$

Under the conditions $s_i^T \mathbf{1} = 1$, we can get:

$$\sum_{j=1}^{k} (\frac{\lambda_1}{2\lambda_2} \hat{Z}_{i,k} + \tau) = 1 \Rightarrow \tau = \frac{1}{k} + \frac{\lambda_1}{2k\lambda_2} \sum_{j=1}^{k} \hat{Z}_{i,k} \tag{20}$$

Update α by fixing A, B and S

After fixing $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{S}$, the objective function (7) becomes:

$$\min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{A}\boldsymbol{B} \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \tag{21}$$

In order to optimize the next step, here is the simplification of the above formula, namely:

$$\begin{aligned} &\Leftrightarrow \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - (\alpha_1 \boldsymbol{K}^{(1)} \boldsymbol{A}\boldsymbol{B} + ... + \alpha_d \boldsymbol{K}^{(d)} \boldsymbol{A}\boldsymbol{B}) \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\overset{\boldsymbol{Q}^{(i)} = \boldsymbol{K}^{(i)} \boldsymbol{A}\boldsymbol{B} \in \boldsymbol{R}^{n \times d}}{\Leftrightarrow} \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{X} - (\alpha_1 \boldsymbol{Q}^{(1)} + ... + \alpha_d \boldsymbol{Q}^{(d)}) \right\|_F^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \\ &\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i - (\alpha_1 q_{i,\cdot}^{(1)} + ... + \alpha_d q_{i,\cdot}^{(d)}) \right\|_2^2 + \lambda_3 \left\| \boldsymbol{\alpha} \right\|_1 \end{aligned} \tag{22}$$

We set $\boldsymbol{M}^{(i)} = \begin{pmatrix} q_{i,1}^{(1)} & q_{i,d}^{(1)} \\ & \\ q_{i,1}^{(d)} & q_{i,d}^{(d)} \end{pmatrix} \in \boldsymbol{R}^{d \times d}$ and have:

$$\begin{aligned}
&\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i - \boldsymbol{\alpha}^T \boldsymbol{M}^{(i)} \right\|_2^2 + \lambda_3 \|\boldsymbol{\alpha}\|_1 \\
&\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \left\| \boldsymbol{X}_i^T - (\boldsymbol{M}^{(i)})^T \boldsymbol{\alpha} \right\|_2^2 + \lambda_3 \|\boldsymbol{\alpha}\|_1 \\
&\Leftrightarrow \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \boldsymbol{X}_i \boldsymbol{X}_i^T - 2\boldsymbol{\alpha}^T \sum_{i=1}^{n} \boldsymbol{M}^{(i)} \boldsymbol{X}_i^T \\
&+ \boldsymbol{\alpha}^T \sum_{i=1}^{n} (\boldsymbol{M}^{(i)} (\boldsymbol{M}^{(i)})^T) \boldsymbol{\alpha} + \lambda_3 \|\boldsymbol{\alpha}\|_1
\end{aligned} \tag{23}$$

The above simplification is only for the convenience of the following gradient descent (Wang *et al.*, 2016). We make:

$$\begin{aligned}
f(\boldsymbol{\alpha}) &= \left\| \boldsymbol{X} - \sum_{i=1}^{d} \alpha_i \boldsymbol{K}^{(i)} \boldsymbol{A} \boldsymbol{B} \right\|_F^2 \\
F(\boldsymbol{\alpha}) &= f(\boldsymbol{\alpha}) + \lambda_3 \|\boldsymbol{\alpha}\|_1
\end{aligned} \tag{24}$$

Note that $\|\boldsymbol{\alpha}\|_1$ is convex but not smooth. So using approximate gradient to optimize $\boldsymbol{\alpha}$, we can update iterations $\boldsymbol{\alpha}$ by the following rules.

$$\boldsymbol{\alpha}_{t+1} = \arg\min_{\boldsymbol{\alpha}} G_{\eta t}(\boldsymbol{\alpha}, \boldsymbol{\alpha}_t) \tag{25}$$

$$G_{\eta t}(\boldsymbol{\alpha}, \boldsymbol{\alpha}_t) = f(\boldsymbol{\alpha}_t) + < \nabla f(\boldsymbol{\alpha}_t), \boldsymbol{\alpha} - \boldsymbol{\alpha}_t > + \frac{\eta_t}{2} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}_t\|^2 + \lambda_3 \|\boldsymbol{\alpha}\|_1 \tag{26}$$

In the above formula, $\nabla f(\boldsymbol{\alpha}_t) = 2\boldsymbol{\alpha}_t^T \sum_{i=1}^{n} (\boldsymbol{M}^{(i)} (\boldsymbol{M}^{(i)})^T) - 2\sum_{i=1}^{n} \boldsymbol{X}_i (\boldsymbol{M}^{(i)})^T$, η_t is a tuning parameter, $\boldsymbol{\alpha}_t$ is the value of $\boldsymbol{\alpha}$ in the *t*-th iteration.

By ignoring the independence in equation (26), we can get:

$$\boldsymbol{\alpha}_{t+1} = \pi_{\eta_t}(\boldsymbol{\alpha}_t) = \arg\min_{\alpha} \frac{1}{2} \|\boldsymbol{\alpha} - \boldsymbol{U}_t\|_F^2 + \frac{\lambda_3}{\eta_t} \|\boldsymbol{\alpha}\|_1 \tag{27}$$

Among them $\boldsymbol{U}_t = \boldsymbol{\alpha}_t - \frac{1}{\eta_t} \nabla f(\boldsymbol{\alpha}_t)$, $\pi_{\eta_t}(\boldsymbol{\alpha}_t)$ is the Euclidean projection on the convex set η_t, because $\|\boldsymbol{\alpha}\|_1$ has a separable form, the formula (27) can be written as follows:

$$\alpha_{t+1}^i = \arg\min_{\alpha^i} \frac{1}{2} \|\alpha^i - U_t^i\|_2^2 + \frac{\lambda_3}{\eta_t} |\alpha^i| \tag{28}$$

Where α^i and α_{t+1}^i are the *i*-th elements of $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_{t+1}$ respectively, then according to formula (28), α_{t+1}^i can obtain the following closed solution:

$$\alpha^{i*} = \begin{cases} u_t^i - \frac{\lambda_3}{\eta_t} \times sign(u_t^i), & \text{if } \|u_t^i\| > \frac{\lambda_3}{\eta_t} \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

To speed up the approximate gradient algorithm in equation (26), we have added auxiliary variables:

$$\boldsymbol{V}_{t+1} = \boldsymbol{\alpha}_t + \frac{\beta_t - 1}{\beta_{t+1}} (\boldsymbol{\alpha}_{t+1} - \boldsymbol{\alpha}_t) \tag{30}$$

Where $\beta_{t+1} = \frac{1 + \sqrt{1 + 4\beta_t^2}}{2}$.

EXPERIMENTS

In this part, we arrange the proposed algorithm and the comparison algorithm in the same environment for experiment, and classify the data after the algorithm is reduced by SVM. Finally, the validity and performance of the feature selection algorithm are measured according to the classification accuracy.

3.1 Experiment Settings

We tested our proposed unsupervised feature selection algorithm on four binary data sets and eight multi-class data sets. They are Yale, Colon, Lung_discrete, Glass, SPECTF, Sonar, Clean, Arrhythmia, Movements, Ecoli, Urban_land and Forest, where the first three data sets are from feature selection data, and the last nine data sets are from the UCI data set. The details of the data set are shown in Table 1:

Table1: The information of the data sets

Datasets	Samples	Dimensions	Classes
Glass	214	9	6
Movements	360	90	15
SPECTF	267	44	2
Ecoli	336	343	8
Sonar	208	60	2

Urban_land	168	147	9
Clean	476	167	2
Forest	325	27	4
Arrhythmia	452	279	13
Colon	62	2000	2
Yale	165	1024	15
Lungdiscrete	73	325	7

At the same time, we found eight representative feature selection comparison algorithms to compare with our proposed algorithm. The main introduction of the algorithm is as follows:

EUFS (Wang *et al.*, 2015): It directly embeds the unsupervised feature selection algorithm into the clustering algorithm through sparse learning, which is an embedded feature selection algorithm. It applies the $l_{2,1}$ – norm to the cost function to reduce the impact of reconstructed data matrices and feature selection on $\boldsymbol{V}$.

FSASL (Du L and Shen Y D, 2015): By classifying the previous unsupervised feature selection algorithm, a novel learning framework is proposed, which is an unsupervised feature selection algorithm for adaptive structure learning, which combines structure learning and feature learning. Finally, the experiment also shows that the algorithm is very effective.

NDFS (Li *et al.*, 2012): The algorithm makes the algorithm select more representative features through the cluster learning and feature selection matrix of the class label. In order to learn more accurate clustering labels, it performs non-negative constraints. The $l_{2,1}$ – norm is also added to remove the effects of redundant features and noise.

NetFS (Li *et al.*, 2016): Embedding a principle method into the representation learning from network structure learning to feature selection is a robust unsupervised feature selection algorithm. The algorithm uses an alternate optimization algorithm to optimize itself and validate its performance through realistic data representation.

RLSR (Chen *et al.*, 2017): The algorithm is a novel semi-supervised feature selection algorithm that evaluates the importance of features by performing a least squares regression by re-adjusting the regression coefficients with a set of scale factors. At the same time, the $l_{2,1}$ – norm was added. The model not only performs global learning, but also performs sparse learning.

RFS (Nie *et al.*, 2010): The algorithm is an efficient and robust feature selection algorithm that applies the $l_{2,1}$ – norm to both the loss function and the regularization term. It can regularize sparsely all data to select more representative features. The algorithm performs better in genomic and proteomic biology and can perform good biological information learning tasks.

RSR (Zhu *et al.*, 2015): The algorithm is an unsupervised feature selection algorithm for self-characterization learning. Each feature is linearly combined by other features, and the $l_{2,1}$ – norm is applied to the coefficient matrix and the residual matrix. It effectively selects features and ensures robustness to outliers. If a feature is important, it will participate in the representation of most other features, resulting in a series of important representation coefficients, and vice versa.

K_OFSD (Zhou *et al.*, 2017): This article defines the online stream feature selection problem of class imbalance data, and proposes an online feature selection framework. The algorithm is based on the dependency between conditional features and decision-making classes, while refining the neighborhood rough set theory and using a fixed number of nearest neighbors of the selected features to solve the class imbalance problem.

In our proposed model, we set $\{\lambda_1, \lambda_2\} \in \{10^{-4}, ..., 10^{8}\}$, the rank of the kernel coefficient matrix $r \in \{1, ..., \min(n, d)\}$, and the parameter of $l_{2,p}$ – norm $p \in \{0.1, ..., 1.9\}$. The parameters $c \in \{2^{-5}, ..., 2^{5}\}$ and $g \in \{2^{-5}, ..., 2^{5}\}$ are used to select the best SVM for classification. Through 10-fold cross-validation, we divide the data set into a training set and a test set. In order to minimize the experimental error, we perform 10 times and 10 folds, and finally find the average of the classification accuracy.

Since the experiment uses the classification accuracy rate to measure the performance of the algorithm, we define the classification accuracy as follows:

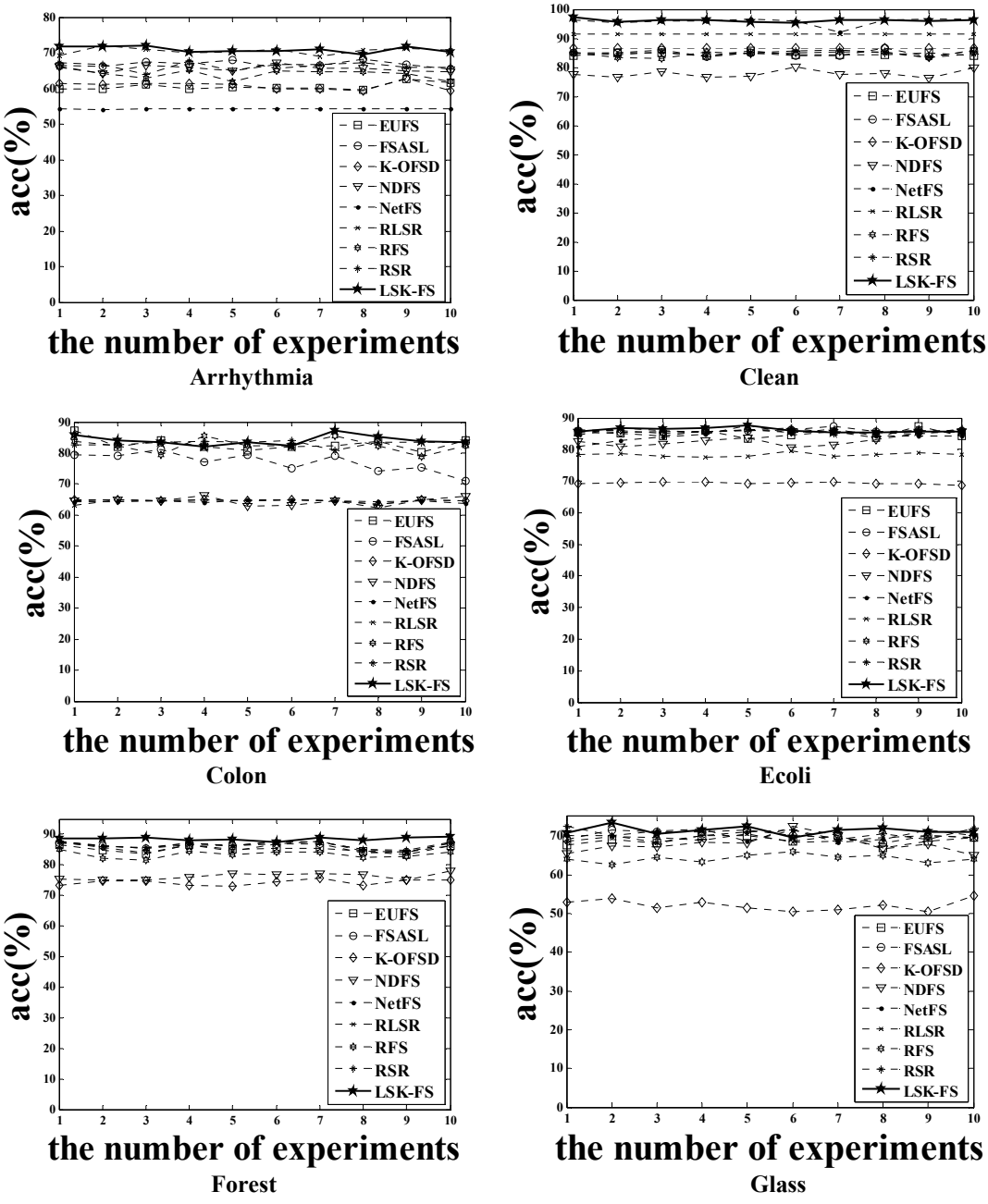
$$acc = X_{correct} / X \tag{31}$$

Where X represents the total number of samples and $X_{correct}$ represents the correct number of samples for classification. At the same time we define the standard deviation to measure the stability of our algorithm, as follows:

$$std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(acc_i - \mu)^2} \tag{32}$$

Where N represents the number of experiments, acc_i represents the classification accuracy of the i-th experiment, μ represents the average classification accuracy, and the smaller the std, the more stable the representative algorithm.

3.2 Experiment Results and Analysis



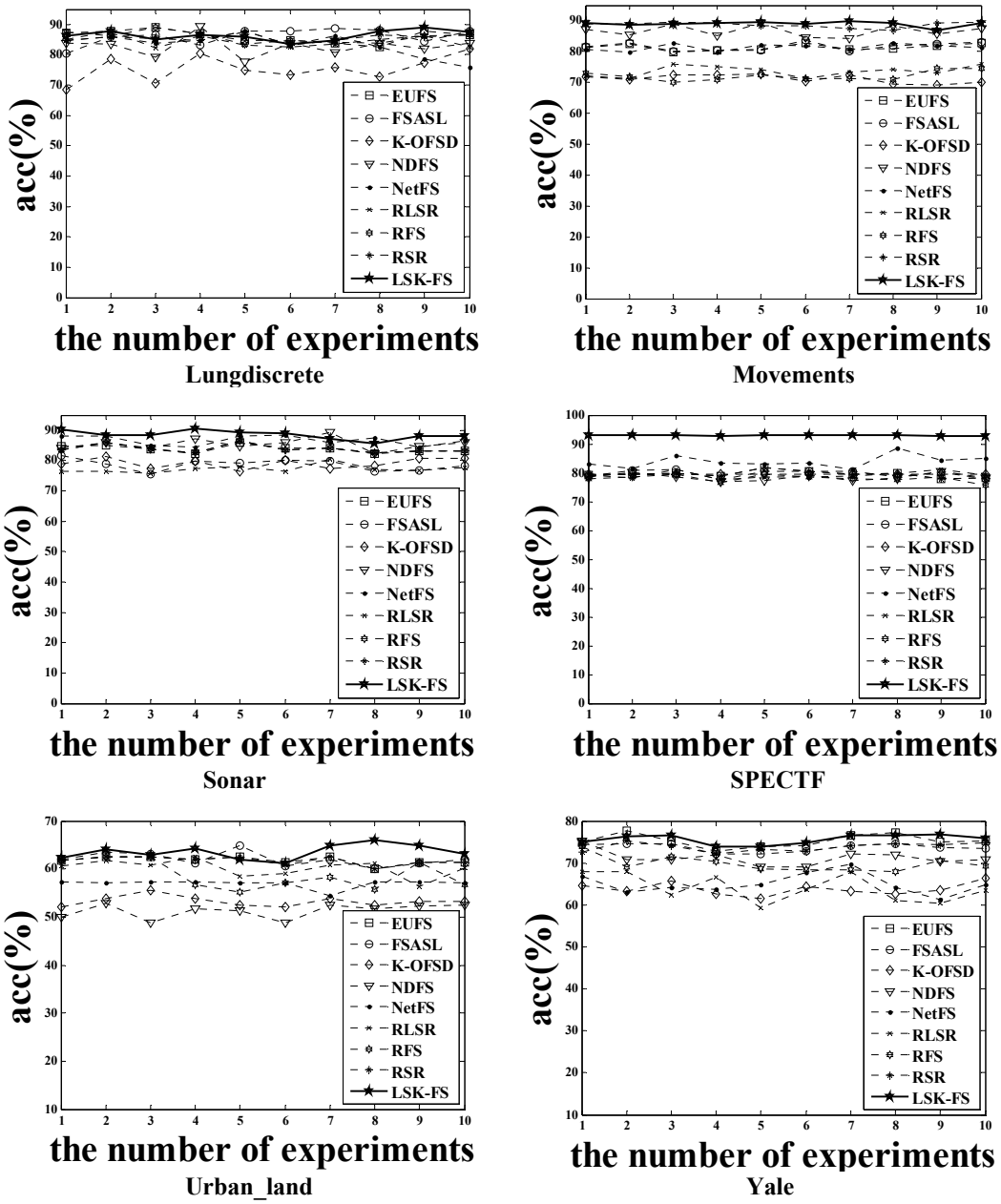


Fig 1 : Average classification accuracy of all methods for all datasets

In Figure 1, we can clearly see the classification accuracy of the 10 experiments. The algorithm we proposed is not the highest every time, but most of the cases are the highest. In Table 2, we can see the average classification accuracy of each algorithm on 12 data sets. The algorithm proposed by us is obviously superior to other comparison algorithms. Specifically, it is 4.78% higher than EUFS in average classification accuracy and 5.05% higher than FSASL, which indicates that our algorithm is better than the general feature selection algorithm. Compared with K_OFSD, NDFS, NetFS, RLSR, RFS, and RSR. LSK_FS increased 13.63%, 8.55%, 6.69%, 7.88%, 6.68%, and 3.59%, respectively. In particular, our algorithm is particularly effective on the dataset SPECTF.

Table2: Average classification accuracy (acc(%))

Datasets	EUFS	FSASL	K_OFSD	NDFS	NetFS	RLSR	RFS	RSR	LSK_FS
Arrhythmia	60.52	66.71	60.74	65.57	54.20	70.53	64..02	66.33	70.96
Clean	84.61	84.83	86.53	77.82	95.78	91.53	84.64	84.77	96.13
Colon	82.88	77.07	64.60	64.43	64.38	64.42	82.48	83.17	84.14
Ecoli	84.97	86.01	69.28	82.81	84.47	78.31	85.74	85.81	86.19

Forest	85.23	86.06	74.23	76.31	86.43	85.78	83.50	86.55	88.50
Glass	68.90	70.17	52.09	67.86	70.20	69.36	64.17	70.42	71.38
Lungdiscrete	86.11	86.07	75.45	82.77	73.41	84.43	85.88	85.55	86.41
Movements	81.36	81.61	71.19	86.61	81.42	73.58	72.08	88.61	88.92
Sonar	83.81	78.67	79.11	85.43	86.66	77.26	83.91	83.91	88.48
SPECTF	79.54	79.70	79.40	78.48	83.95	79.22	78.04	79.54	92.94
Urban_land	61.78	62.05	53.25	51.27	56.96	60.20	58.82	61.72	63.56
Yale	75.27	73.69	63.78	71.22	65.04	64.05	69.76	73.81	75.63
Average value	77.92	77.72	69.14	74.22	76.08	74.89	76.09	79.18	82.77

In Table 3, we can see the average standard deviation of each algorithm on 12 data sets. The standard deviation of the proposed LSK_FS algorithm is the smallest, indicating that our algorithm has the best stability.

Table3: Standard deviation of classification accuracy (std(%))

Datasets	EUFS	FSASL	K_OFSD	NDFS	NetFS	RLSR	RFS	RSR	LSK_FS
Arrhythmia	0.88	0.85	1.04	0.89	0.02	0.85	1.29	1.09	0.79
Clean	0.54	0.94	0.03	1.29	1.32	0.03	0.97	0.74	0.49
Colon	1.88	2.99	0.38	1.31	0.29	0.59	2.08	0.93	1.5
Ecoli	1.18	0.47	0.30	1.59	1.78	0.55	0.40	0.82	0.69
Forest	1.05	1.05	0.93	1.05	1.06	1.29	1.16	0.99	0.54
Glass	1.05	1.02	1.37	2.07	1.15	0.70	1.00	1.29	1.01
Lungdiscrete	1.92	2.52	4.07	3.03	3.43	1.82	1.67	1.21	1.60
Movements	1.04	1.31	1.33	1.76	1.06	1.65	1.44	0.80	0.72
Sonar	1.18	1.81	1.62	1.85	1.45	1.19	1.28	1.28	1.38
SPECTF	1.01	1.24	0.03	1.20	2.08	1.10	1.11	1.01	0.04
Urban_land	0.77	1.38	1.01	1.44	0.88	1.69	2.74	0.83	1.41
Yale	1.44	0.95	1.4	1.71	2.28	3.16	1.74	0.99	1.08
Average value	1.16	1.38	1.13	1.60	1.40	1.22	1.41	1.00	0.94

The LSK_FS algorithm can achieve such good results, mainly for the following two reasons: 1. Consider the similarity between data features. 2. Fully consider the nonlinear relationship between data features.

CONCLUSION

In this paper, a new unsupervised nonlinear feature selection algorithm is proposed by considering the similarity and nonlinear relationship between data features. That is, the local structure learning is used to find the similarity between the features, then the kernel method is used to find the nonlinear relationship between the data features, and finally the feature selection is performed by the sparse regularization factor. Low rank constraints have also been added to the model to better refine the proposed model. More significant mining results than the general feature selection algorithm. The experimental results show that the proposed algorithm has achieved great improvement in classification accuracy and stability. In the future work, we attempt to combine more advanced theoretical improvement algorithms.

ACKNOWLEDGMENTS

This work is partially supported by the China Key Research Program (Grant No: 2016YFB1000905); the Key Program of the National Natural Science Foundation of China (Grant No: 61836016); the Natural Science Foundation of China (Grants No: 61876046, 61573270, 81701780 and 61672177); the Project of Guangxi Science and Technology (GuiKeAD17195062); the Guangxi Natural Science Foundation (Grant No: 2015GXNSFCB139011, 2017GXNSFBA198221); the Guangxi Collaborative Innovation Center of Multi-Source Information Integration and Intelligent Processing; the Guangxi High Institutions Program of Introducing 100 High-Level Overseas Talents; and the Research Fund of Guangxi Key Lab of Multisource Information Mining and Security (18-A-01-01).

REFERENCES

[1] Zhu X, Suk H I, Shen D. (2014). Matrix-Similarity Based Loss Function and Feature Selection for Alzheimer's Disease Diagnosis. Computer Vision and Pattern Recognition. IEEE, 2014:3089-3096.

[2] Zhu X, Huang Z, Yang Y, *et al.* (2013). Self-taught dimensionality reduction on the high-dimensional small-sized data. Pattern Recognition, 2013, 46(1):215-229.

[3] Zhu X, Zhang S, Jin Z, *et al.* (2010). Missing Value Estimation for Mixed-Attribute Data Sets. IEEE Transactions on Knowledge & Data Engineering, 2010, 23(1):110-121.

[4] Bolón-Canedo V, Sánchez-Maroño N, Alonso-Betanzos A. (2016) Feature selection for high-dimensional data. Progress in Artificial Intelligence, 2016, 5(2):1-11.

[5] Ling C X, Yang Q, Wang J, *et al.* (2004). Decision trees with minimal costs. International Conference on Machine Learning. ACM, 2004:69.
[6] Zhu X, Huang Z, Yang Y, *et al.* (2013). Self-taught dimensionality reduction on the high-dimensional small-sized data. Pattern Recognition, 2013, 46(1):215-229.
[7] Zhang S, Jin Z, Zhu X. (2011). Missing data imputation by utilizing information within incomplete instances. Journal of Systems & Software, 2011, 84(3):452-459.
[8] Nie F, Zhu W, Li X. (2016). Unsupervised feature selection with structured graph optimization. Thirtieth AAAI Conference on Artificial Intelligence. AAAI Press, 2016:1302-1308.
[9] Almusallam N, Tari Z, Chan J, *et al.* (2018). UFSSF - An Efficient Unsupervised Feature Selection for Streaming Features. 2018, 1093(20):495-507.
[10] Xue W, Zhang W. (2016). Online Weighted Multi-task Feature Selection. Neural Information Processing. Springer International Publishing, 2016:195-203.
[11] Liu X, Wang L, Zhang J, *et al.* (2017). Global and Local Structure Preservation for Feature Selection. IEEE Transactions on Neural Networks & Learning Systems, 2017, 25(6):1083-1095.
[12] Wan Y, Chen X, Zhang J. (2018). Global and Intrinsic Geometric Structure Embedding for Unsupervised Feature Selection. Expert Systems with Applications, 2018, 93(1):134-142.
[13] Li Y, Si J, Zhou G, *et al.* (2017). FREL: A Stable Feature Selection Algorithm. IEEE Transactions on Neural Networks & Learning Systems, 2017, 26(7):1388-1402.
[14] Tsagris M, Lagani V, Tsamardinos I. (2018). Feature selection for high-dimensional temporal data. Bmc Bioinformatics, 2018, 19(1):17.
[15] Zhao H, Wang P, Hu Q. (2016). Cost-sensitive feature selection based on adaptive neighborhood granularity with multi-level confidence. Information Sciences, 2016, 366(20):134-149.
[16] Wang W, Yan Y, Winkler S, *et al.* (2016). Category Specific Dictionary Learning for Attribute Specific Feature Selection. IEEE Transactions on Image Processing, 2016, 25(3):1465-1478.
[17] Sheeja T K, Kuriakose A S. (2018). A novel feature selection method using fuzzy rough sets. Computers in Industry, 2018, 97(5):111-121.
[18] Zhang T, Ding B, Zhao X, *et al.* (2018). A Fast Feature Selection Algorithm Based on Swarm Intelligence in Acoustic Defect Detection. IEEE Access, 2018, 6(99):28848-28858.
[19] Zhang C, Qin Y, Zhu X, *et al.* (2007). Clustering-based Missing Value Imputation for Data Preprocessing. IEEE International Conference on Industrial Informatics. IEEE, 2007:1081-1086.
[20] Qin Y, Zhang S, Zhu X, *et al.* (2007). Semi-parametric optimization for missing data imputation. Applied Intelligence, 2007, 27(1):79-88.
[21] Wang S, Tang J, and Liu H. (2015) Embedded Unsupervised Feature Selection. AAAI. 2015: 470-476.
[22] Du L and Shen Y D. (2015) Unsupervised feature selection with adaptive structure learning. Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, 2015: 209-218.
[23] Li Z, Yang Y, Liu J, *et al.* (2012) Unsupervised feature selection using nonnegative spectral analysis. AAAI. 2012, 2: 1026-1032.
[24] Li J, Hu X, Wu L, *et al.* (2016) Robust unsupervised feature selection on networked data. Proceedings of the 2016 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2016: 387-395.
[25] Chen X, Yuan G, Nie F, *et al.* (2017) Semi-supervised Feature Selection via Rescaled Linear Regression. IJCAI. 2017: 1525-1531.
[26] Nie F, Huang H, Cai X, *et al.* (2010) Efficient and robust feature selection via joint ℓ2, 1-norms minimization. Advances in neural information processing systems. 2010: 1813-1821.
[27] Zhu P, Zuo W, Zhang L, *et al.* (2015) Unsupervised feature selection by regularized self-representation. Pattern Recognition, 2015, 48(2): 438-446.
[28] Zhou P, Hu X, Li P, *et al.* Online feature selection for high-dimensional class-imbalanced data. Knowledge-Based Systems, 2017, 136(15): 187-199.