City Research Online

City, University of London Institutional Repository

Citation: Wang, X., Zhu, R. & Xue, J-H. (2025). GKF-PUAL: A group kernel-free approach to positive-unlabeled learning with variable selection. Information Sciences, 690, 121574. doi: 10.1016/j.ins.2024.121574

This is the published version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/33901/

Link to published version: https://doi.org/10.1016/j.ins.2024.121574

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Information Sciences 690 (2025) 121574

Contents lists available at ScienceDirect

Information Sciences

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

GKF-PUAL: A group kernel-free approach to positive-unlabeled learning with variable selection

Xiaoke Wang [a], Rui Zhu [b,*], Jing-Hao Xue [a]

[a] *Department of Statistical Science, University College London, Gower Street, London, WC1E 6BT, England, UK*
[b] *Bayes Business School, City St George's, University of London, 106 Bunhill Row, London, EC1Y 8TZ, England, UK*

A R T I C L E I N F O

Keywords:
Positive-unlabeled learning
Group lasso
Kernel-free approach
Trifurcate data
Variable selection

A B S T R A C T

Variable selection is important for classification of data with many irrelevant predicting variables, but it has not yet been well studied in positive-unlabeled (PU) learning, where classifiers have to be trained without labelled-negative instances. In this paper, we propose a group kernel-free PU classifier with asymmetric loss (GKF-PUAL) to achieve quadratic PU classification with group-lasso regularisation embedded for variable selection. We also propose a five-block algorithm to solve the optimization problem of GKF-PUAL. Our experimental results reveal the superiority of GKF-PUAL in both PU classification and variable selection, improving the baseline PUAL by more than 10% in F1-score across four benchmark datasets and removing over 70% of irrelevant variables on six benchmark datasets. The code for GKF-PUAL is at https://github.com/tkks22123/GKF-PUAL.

1. Introduction

Variable selection, also called feature selection, is an important and widely-studied topic in statistics and machine learning for supervised classification [2,4,13,29,32], as irrelevant variables can often damage the interpretability and generalizability of classification models. Models focusing on the most predictive variables usually generalize better on unseen data. Additionally, having fewer variables included can offer easier interpretation of the model; this is especially important in areas such as finance and healthcare where interpretability is very much valued by professionals. There are mainly three ways to achieve variable selection: filter, wrapper, and embedded methods. Filter methods evaluate the relevance of variables to classification based on some metrics and then eliminate those variables with low relevance. Wrapper methods evaluate variable subsets based on the model performance and then select the best-performing subset. Embedded methods, such as lasso, aim to exploit the best of both filter methods and wrapper methods, by embedding variable selection into the model training. Hence, compared with filter methods and wrapper methods, embedded methods offer a better balance between computational complexity and variable-selection accuracy to the classifier training. However, for positive-unlabeled (PU) learning, variable selection, especially embedded variable selection, has not yet been well studied.

PU learning has a variety of applications in practice; for example, learning the political interests of Members of Parliament to assist document distribution [9], time series classification [10,15] and learning to rank in recommendation systems [18]. However, PU learning is a much harder task than supervised classification, because in PU learning only some positive instances are labelled before the model training, i.e. there is no labelled-negative instances available in the training set of PU learning.

* Corresponding author.
E-mail address: rui.zhu@city.ac.uk (R. Zhu).

https://doi.org/10.1016/j.ins.2024.121574
Received 15 June 2024; Received in revised form 9 September 2024; Accepted 19 October 2024

Available online 22 October 2024
0020-0255/© 2024 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

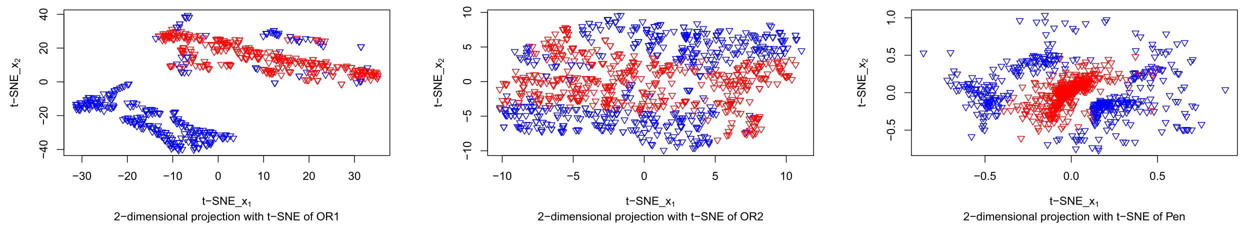

Fig. 1. The t-SNE plots of the three trifurcate datasets **OR1**, **OR2** and **Pen**; blue: positive instances; red: negative instances. The positive instances roughly scatter on both sides of the negative instances.

Therefore, it is quite natural to develop a two-step approach to PU learning [8,16,17,25,34], which involves firstly a method to identify some reliably negative instances from the unlabeled set, and then a semi-supervised method trained on three sets: the set of labelled-positive instances, the set of reliably negative instances as labelled negative instances, and the set of remaining unlabeled instances. However, the performance of a two-step method heavily relies on the quality of the identified reliably negative instances in the first step. Hence, many one-step methods for PU learning, without firstly identifying reliably negative instances, have been proposed, e.g., unbiased PU learning (uPU) [11,12], non-negative PU learning (nnPU) [22], imbalanced nnPU [31], cost-sensitive PU learning [7], biased support vector machine [24], weighted unlabeled samples SVM [26], biased least squares SVM [21], GLLC [20], and PUAL [33].

As we can imagine, the unavailability of labelled-negative instances for the model training also makes variable selection a more challenging and less-studied task for PU learning than for supervised classification. [5] provided a pioneer method for variable selection in PU learning called puCFS, which is a filter method; however, the class prior is a hyper-parameter of this method and has to be set/estimated before variable selection; [19] evaluated some unsupervised filter methods for variable selection in PU learning. [30] built a robust wrapper method to maximise the area under curve (AUC) for the training of a linear classifier for PU learning; [14] proposed an embedded variable-selection method by combining L_1-norm and logistic regression to generate a linear decision boundary for classification on PU data. However, these two methods mainly work on the classifiers generating linear decision boundary.

We build our model on PUAL [33], a PU classifier particularly effective on the datasets having trifurcate patterns as illustrated in Fig. 1 by using three UCI datasets (UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/index.php), as such a pattern necessitates a non-linear classifier. The kernel trick is widely adopted by classifiers to generate nonlinear decision boundary. However, we note that the kernel trick is not compatible to the lasso or group-lasso regularisation term for embedding variable selection in PUAL. To resolve this issue, we propose a group kernel-free quadratic PU classifier (GKF-PUAL) with the ability of embedded variable selection in this paper.

Technically more specific, to construct GKF-PUAL, we firstly introduce the quadratic predictive score and objective functions of a kernel-free SVM [27] to PUAL to generate quadratic decision boundaries without relying on the kernel trick. Then, we incorporate the group-lasso regularisation [28,35] into the new objective function of PUAL, aiming to compress the coefficients of irrelevant variables to zero for variable selection. With these two improvements, we build GKF-PUAL on PUAL; and GKF-PUAL demonstrates superior classification and variable-selection performance on PU data (especially trifurcate PU data).

In summary, the technical novelties and contributions of this paper are three-fold:

1. In Section 3 we propose GKF-PUAL as an integration of PUAL, kernel-free SVM, and group lasso to generate quadratic classification with variable selection embedded.
2. We propose a five-block algorithm in Sections 3 to solve the optimization problem of GKF-PUAL.
3. Experiments in Section 4 on the real-world datasets verify the superiority of GKF-PUAL in classification and variable selection of PU data.

2. Related work

We first establish some notation. Suppose there are n_p labelled-positive instances and n_u unlabeled instances with m predicting variables (aka features). Then $\boldsymbol{X}_{[pu]} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n_p}, \dots, \boldsymbol{x}_{n_p+n_u})^T \in \mathbb{R}^{(n_p+n_u)\times m}$ is the matrix with vector $\boldsymbol{x}_i \in \mathbb{R}^{m\times 1}$ to be the vector of predicting variables of the ith instance, and $n_p + n_u = n$. Similarly, $\boldsymbol{X}_{[p]} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n_p})^T \in \mathbb{R}^{n_p\times m}$ is the matrix for the labelled-positive instances and $\boldsymbol{X}_{[u]} = (\boldsymbol{x}_{n_p+1}, \dots, \boldsymbol{x}_{n_p+n_u})^T \in \mathbb{R}^{n_u\times m}$ is the matrix for the unlabeled instances including both unlabeled positive and negative instances.

2.1. PUAL [33]

For better PU classification performance, especially on the trifurcate data like those illustrated in Fig. 1, PUAL [33] was proposed by replacing the squared loss in the objective function of GLLC [20] with the hinge loss for labelled-positive instances, thus providing an asymmetric structure of loss able to treat labelled-positive instances and unlabeled-positive instances differently.

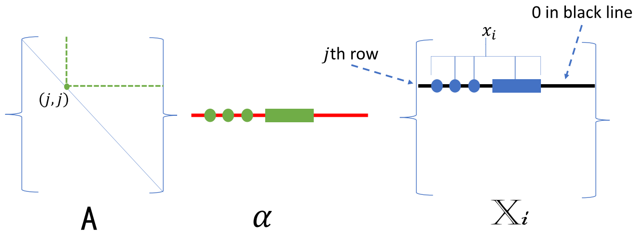

Fig. 2. Illustration of the steps to construct the jth row of matrix $\mathbb{X}_i$: the elements marked with green circles in vector $\boldsymbol{\alpha}$ are equivalent to those represented by the green dashed lines in matrix **A**; the elements marked by the blue circles constitute $\boldsymbol{x}_i$, and their positions in the jth row of matrix $\mathbb{X}_i$ correspond to the same positions as the elements marked by the green circles in vector $\boldsymbol{\alpha}$.

The linear predictive score function of PUAL is simple as

$$f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\beta} + \beta_0, \tag{1}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m)^T \in \mathbb{R}^{m\times 1}$ and β_0 are the model parameters to learn.

The objective function of PUAL can be expressed as

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} & \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + C_p\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ \\ & + C_u[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]^T[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)] \\ & + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0),\end{aligned} \tag{2}$$

where C_p, C_u and $\lambda > 0$ are hyper-parameters of model, $\mathbf{1}_{p,u} = \underbrace{(1, 1, \cdots, 1)}_{k}^T$ with $k = n_p$ or n_u, and $\boldsymbol{R}$ denotes a matrix of local similarity between instances based on their K-nearest neighbours and a hyper-parameter σ (See the details of calculation of $\boldsymbol{R}$ in [20,33]), and $[g(\cdot)]_+$ is a column vector of the maximum between each element of $g(\cdot)$ and 0.

To generate non-linear decision boundary for PUAL, the kernel trick can be adopted. However, the kernel trick cannot be applied to the objective function of PUAL if an additional lasso or group-lasso regularisation term is introduced for variable selection. This is because in that case the terms of the feature maps (from the original feature space to an infinite-dimensional space) cannot be rewritten in the form of self-inner products for kernel matrices to replace them.

2.2. Soft quadratic surface support vector machine (SQS-SVM)

As a kernel-free SVM, SQS-SVM [27] is a classifier holding the following quadratic predictive score function

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\mathbf{A}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{\beta} + \beta_0, \tag{3}$$

where the symmetric matrix $\mathbf{A} = \{\alpha_{ij}\}, i, j = 1, \ldots, m$ is an $m \times m$ matrix of the quadratic model parameters to be learnt. We note that GKF-PUAL, which is to be proposed in this paper, also uses the same predictive score function as that of SQS-SVM in Equation (3).

To train SQS-SVM, the objective function can be formulated as

$$\min_{\mathbf{A},\boldsymbol{\beta},\beta_0} \frac{\lambda_2}{2}\sum_{i=1}^{n}\|\mathbf{A}\boldsymbol{x}_i + \boldsymbol{\beta}\|_2^2 + \sum_{i=1}^{n}\left[1 - y_i(\frac{1}{2}\boldsymbol{x}_i^T\mathbf{A}\boldsymbol{x}_i + \boldsymbol{x}_i^T\boldsymbol{\beta} + \beta_0)\right]_+, \tag{4}$$

where n is the size of the training set and y_i is the ground-truth class label of the ith instance. To solve this optimization problem, [27] offers a way to first transform Equation (4) into a form similar to the objective function of classic supervised SVM, by converting matrix **A** to a vector, briefly described as follows.

Firstly, place the upper triangle elements of matrix **A** into $\frac{m^2+m}{2}$-dimensional vector

$$\boldsymbol{\alpha} = (\alpha_{11}, \alpha_{12}, \ldots \alpha_{1m}, \alpha_{22}, \ldots \alpha_{2m}, \ldots, \alpha_{mm})^T. \tag{5}$$

Then, map the ith independent variable $\boldsymbol{x}_i$ into an $m \times \frac{m^2+m}{2}$ matrix $\mathbb{X}_i$ in the following steps, which is illustrated in Fig. 2:

1. Initialise $\mathbb{X}_i$ as a $\{\mathbf{0}\}_{m\times\frac{m^2+m}{2}}$ matrix.
2. Record the coordinate g of the element in vector $\boldsymbol{\alpha}$ if it is originally from the jth row or column of matrix **A**. The other coordinate than j of this element in matrix **A** is subsequently recorded as k.
3. The gth element in the jth row of matrix $\mathbb{X}_i$ is determined as $\boldsymbol{x}_{ik}$.
4. Repeat Step 2 and Step 3 until the position of all the elements in vector $\boldsymbol{\alpha}$ originally from the jth row or column of matrix **A** is recorded for a fixed j.

5. Repeat Step 4 until j has taken the value from 1 to m.
6. Repeat Step 5 until i has taken the value from 1 to n.

After obtaining $\boldsymbol{\alpha}$ and $\mathbb{X}_i, i = 1, \ldots, n$, and by defining $\boldsymbol{\alpha}^* = [\boldsymbol{\alpha}^T, \boldsymbol{\beta}^T]^T$ and $\mathbb{X}_i^* = [\mathbb{X}_i, \boldsymbol{I}_m]$, then the objective function of SQS-SVM can be reformulated as

$$\min_{\boldsymbol{\alpha}^*, \beta_0} \frac{\lambda_2}{2} \boldsymbol{\alpha}^{*T} \boldsymbol{G} \boldsymbol{\alpha}^* + \mathbf{1}_n^T \left[\mathbf{1}_n - (\boldsymbol{S}\boldsymbol{\alpha}^* + \mathbf{1}_n \beta_0) \right]_+, \tag{6}$$

where $\boldsymbol{G} = \sum_{i=1}^n \left(\mathbb{X}_i^*\right)^T \mathbb{X}_i^*$, $\boldsymbol{s}_i = \frac{1}{2}[\mathbb{X}_i^{*T} \boldsymbol{x}_i + (\mathbf{0}^T_{\frac{m^2+m}{2}}, \boldsymbol{x}_i^T)^T]$, $\boldsymbol{S} = \{\boldsymbol{s}_i^T\}_{n \times m^*}$, and $m^* = \frac{m^2+3m}{2}$.

3. GKF-PUAL

3.1. Objective function

To propose the objective function of GKF-PUAL, we adopt two steps: 1) first modify the objective function of linear PUAL in Equation (2) to obtain kernel-free quadratic PUAL; and 2) then introduce a group-lasso regularisation to the kernel-free quadratic PUAL for variable selection.

3.1.1. Kernel-free quadratic PUAL

Firstly, we consider an objective function similar to the objective function of PUAL in Equation (2), but we replace the linear terms of $\boldsymbol{X}$ in Equation (2) by their quadratic counterparts and replace the L_2-norm of $\boldsymbol{\beta}$ in Equation (2) by the corresponding regularisation term of SQS-SVM in Equation (4):

$$\begin{aligned}
\min_{\mathbf{A}, \boldsymbol{\beta}, \beta_0} & \frac{\lambda_2}{2} \sum_{i=1}^{n_{pu}} \|\mathbf{A}\boldsymbol{x}_i + \boldsymbol{\beta}\|_2^2 + C_p \sum_{i=1}^{n_p} \left[1 - (\frac{1}{2}\boldsymbol{x}_i^T \mathbf{A} \boldsymbol{x}_i + \boldsymbol{x}_i^T \boldsymbol{\beta} + \beta_0)\right]_+ \\
& + C_u \sum_{i=n_p+1}^{n_{pu}} \left[1 + (\frac{1}{2}\boldsymbol{x}_i^T \mathbf{A} \boldsymbol{x}_i + \boldsymbol{x}_i^T \boldsymbol{\beta} + \beta_0)\right]^2 + \boldsymbol{b}^T \boldsymbol{R} \boldsymbol{b} \\
& s.t. \;\; b_i = \frac{1}{2}\boldsymbol{x}_i^T \mathbf{A} \boldsymbol{x}_i + \boldsymbol{x}_i^T \boldsymbol{\beta} + \beta_0, i = 1, \ldots, n_{pu}.
\end{aligned} \tag{7}$$

To solve this optimization problem, we follow the strategy in SQS-SVM as discussed in Section 2.2, by converting $\mathbf{A}$ to a vector $\boldsymbol{\alpha}$. Then terms involving $\mathbf{A}$ can be written as

$$\begin{aligned}
\sum_{i=1}^{n_{pu}} \|\mathbf{A}\boldsymbol{x}_i + \boldsymbol{\beta}\|_2^2 &= \sum_{i=1}^{n_{pu}} \|\mathbb{X}_i \boldsymbol{\alpha} + \boldsymbol{\beta}\|_2^2 = \sum_{i=1}^{n_{pu}} \left\| [\mathbb{X}_i, \boldsymbol{I}_m] \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \right\|_2^2 \\
&= \sum_{i=1}^{n_{pu}} \|\mathbb{X}_i^* \boldsymbol{\alpha}^*\|_2^2 = \boldsymbol{\alpha}^{*T} \boldsymbol{G} \boldsymbol{\alpha}^*,
\end{aligned} \tag{8}$$

where $\boldsymbol{\alpha}^* = [\boldsymbol{\alpha}^T, \boldsymbol{\beta}^T]^T$, $\mathbb{X}_i^* = [\mathbb{X}_i, \boldsymbol{I}_m]$ and $\boldsymbol{G} = \sum_{i=1}^{n_{pu}} \left(\mathbb{X}_i^*\right)^T \mathbb{X}_i^*$, and

$$\begin{aligned}
\frac{1}{2}\boldsymbol{x}_i^T \mathbf{A} \boldsymbol{x}_i + \boldsymbol{x}_i^T \boldsymbol{\beta} + \beta_0 &= \frac{1}{2}\boldsymbol{x}_i^T \mathbb{X}_i \boldsymbol{\alpha} + \boldsymbol{x}_i^T \boldsymbol{\beta} + \beta_0 \\
&= [\frac{1}{2}\boldsymbol{x}_i^T \mathbb{X}_i, \boldsymbol{x}_i^T] \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} + \beta_0 \\
&= s_i^T \boldsymbol{\alpha}^* + \beta_0,
\end{aligned} \tag{9}$$

where $\boldsymbol{s}_i = \frac{1}{2}\left(\mathbb{X}_i^{*T} \boldsymbol{x}_i + [\mathbf{0}^T_{\frac{m^2+m}{2}}, \boldsymbol{x}_i^T]^T \right)$ and $\mathbf{0}_{\frac{m^2+m}{2}}$ is a $(\frac{m^2+m}{2} \times 1)$-dimensional column vector of zeros.

Thus, the objective function in Equation (7) can be transformed to

$$\begin{aligned}
\min_{\boldsymbol{\alpha}^*, \beta_0} & \frac{\lambda_2}{2} \boldsymbol{\alpha}^{*T} \boldsymbol{G} \boldsymbol{\alpha}^* + \boldsymbol{b}^T \boldsymbol{R} \boldsymbol{b} + C_p \mathbf{1}_p^T \left[\mathbf{1}_p - (\boldsymbol{S}_{[p]} \boldsymbol{\alpha}^* + \mathbf{1}_p \beta_0) \right]_+ \\
& + C_u \left[\mathbf{1}_u + (\boldsymbol{S}_{[u]} \boldsymbol{\alpha}^* + \mathbf{1}_u \beta_0) \right]^T \left[\mathbf{1}_u + (\boldsymbol{S}_{[u]} \boldsymbol{\alpha}^* + \mathbf{1}_u \beta_0) \right] \\
& s.t. \; \boldsymbol{b} = \boldsymbol{S}_{[pu]} \boldsymbol{\alpha}^* + \mathbf{1}_{pu} \beta_0,
\end{aligned} \tag{10}$$

where $\boldsymbol{S}_{[pu]} = \{\boldsymbol{s}_i^T\}$ for $i = 1, \ldots, n$, $\boldsymbol{S}_{[p]} = \{\boldsymbol{s}_i^T\}$ for $i = 1, \ldots, n_p$, and $\boldsymbol{S}_{[u]} = \{\boldsymbol{s}_i^T\}$ for $i = n_{p+1}, \ldots, n$.

3.1.2. Introduction of group-lasso regularisation for variable selection

Now we introduce a group-lasso regularisation to the objective function in Equation (10), to generate sparse coefficient vector $\boldsymbol{\alpha}^*$. The reason why here we adopt group lasso rather than lasso is because we need to enforce a group-wise penalty; that is, once a coefficient associated with a variable is compressed to zero, all other coefficients associated with this variable should also be compressed to zero so that this variable can be removed from the model [28]. Hence we propose the objective function of GKF-PUAL as

$$\begin{aligned}
\min_{\boldsymbol{\alpha}^*,\beta_0} \ & \frac{\lambda_2}{2}\boldsymbol{\alpha}^{*T}\boldsymbol{G}\boldsymbol{\alpha}^* + \boldsymbol{b}^T\boldsymbol{R}\boldsymbol{b} + C_p\mathbf{1}_p^T\left[\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0)\right]_+ \\
& + C_u\left[\mathbf{1}_u + (\boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0)\right]^T\left[\mathbf{1}_u + (\boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0)\right] \\
& + \frac{\lambda_1}{2}\sum_{i=1}^m \|\boldsymbol{\alpha}^*_{[i]}\|_2 \\
& s.t. \ \ \boldsymbol{b} = \boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0,
\end{aligned} \tag{11}$$

where the column vector $\boldsymbol{\alpha}^*_{[i]}$, $i = 1,\dots,m$, is the ith subset of $\boldsymbol{\alpha}^*$ containing the coefficients associated with the ith variable. The cardinality of subset $\boldsymbol{\alpha}^*_{[i]}$ fo $i = 1,\dots,m$ is $m+1$ since there is one overlap element between $\boldsymbol{\alpha}^*_{[i]}$ and $\boldsymbol{\alpha}^*_{[j]}$, $\forall i \neq j$.

3.2. Parameter estimation

3.2.1. Five-block ADMM for the optimization of GKF-PUAL

It should be noted that one element in $\boldsymbol{\alpha}^*$ may be associated with up to two variables, therefore there exist group overlaps among $\boldsymbol{\alpha}^*_{[1]}, \boldsymbol{\alpha}^*_{[2]}, \dots, \boldsymbol{\alpha}^*_{[m]}$, i.e., $\boldsymbol{\alpha}^*_{[i]} \cap \boldsymbol{\alpha}^*_{[j]} \neq \emptyset, \forall i, j = 1, 2, \dots, m$. This causes difficulty in ADMM for optimization, so firstly we consider representing the objective function by five blocks (terms) and optimize the parameters in them separately:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}^*,\beta_0,\boldsymbol{h},\boldsymbol{a},\boldsymbol{t},\boldsymbol{b}} \ & \frac{\lambda_1}{2}\sum_{i=1}^m \|\boldsymbol{t}_{[i]}\|_2 + \frac{\lambda_2}{2}\boldsymbol{\alpha}^{*T}\boldsymbol{G}\boldsymbol{\alpha}^* + \boldsymbol{b}^T\boldsymbol{R}\boldsymbol{b} + C_p\mathbf{1}_p^T[\boldsymbol{h}]_+ + C_u\boldsymbol{a}^T\boldsymbol{a} \\
s.t. \ \ & \boldsymbol{h} = \mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0), \\
& \boldsymbol{a} = \mathbf{1}_u + (\boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0), \\
& \boldsymbol{t} = \boldsymbol{D}^*\boldsymbol{\alpha}^*, \\
& \boldsymbol{b} = \boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0,
\end{aligned} \tag{12}$$

where $\boldsymbol{D}^*$ is the $m(m+1) \times \frac{m^2+3m}{2}$ design matrix. More specifically, to construct $\boldsymbol{D}^*$, firstly we initialise it as an $m(m+1) \times \frac{m^2+3m}{2}$ matrix of zeros; then if the jth element of $\boldsymbol{\alpha}^*_{[i]}$ is $\boldsymbol{\alpha}^*_k$, the $(j + (i-1)(m+1), k)$ element of $\boldsymbol{D}^*$ will be set to 1; after repeating this step for all $i = 1,\dots,m$ and $j = 1,\dots,m+1$, we finally obtain $\boldsymbol{D}^*$ in Equation (12). Hence the $m(m+1)$-dimensional column vector $\boldsymbol{t} = (\boldsymbol{t}^T_{[1]}, \boldsymbol{t}^T_{[2]}, \dots, \boldsymbol{t}^T_{[m]})^T$, and the cardinality of $\boldsymbol{t}_{[i]}$, $i = 1,\dots,m$, is $m+1$. The introduced local variable $\boldsymbol{t}$ can eliminate the group overlaps among the group-lasso regularisation terms, i.e., $\forall i \neq j, \boldsymbol{t}_{[i]} \cap \boldsymbol{t}_{[j]} = \emptyset$.

The Lagrangian formulation of Equation (12) can be written as

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}) = & \frac{\lambda_1}{2}\sum_{i=1}^m \|\boldsymbol{t}_{[i]}\|_2 + \frac{\lambda_2}{2}\boldsymbol{\alpha}^{*T}\boldsymbol{G}\boldsymbol{\alpha}^* + \boldsymbol{b}^T\boldsymbol{R}\boldsymbol{b} + C_p\mathbf{1}_p^T[\boldsymbol{h}]_+ + C_u\boldsymbol{a}^T\boldsymbol{a} \\
& + \boldsymbol{u}_{\boldsymbol{h}}^T(\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0) - \boldsymbol{h}) + \boldsymbol{q}^T(\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0 - \boldsymbol{b}) \\
& + \boldsymbol{u}_{\boldsymbol{a}}^T(\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0 - \boldsymbol{a}) + \boldsymbol{v}^T(\boldsymbol{D}^*\boldsymbol{\alpha}^* - \boldsymbol{t})
\end{aligned} \tag{13}$$

where $\boldsymbol{u}_h \in \mathbb{R}^{n_p\times 1}$, $\boldsymbol{u}_a \in \mathbb{R}^{n_u\times 1}$, $\boldsymbol{v} \in \mathbb{R}^{m(m+1)\times 1}$, $\boldsymbol{q} \in \mathbb{R}^{(n_p+n_u)\times 1}$ are dual variables.

Then, the augmented Lagrangian function is

$$\begin{aligned}
\mathcal{L}_a(\boldsymbol{\theta}) = & \mathcal{L}(\boldsymbol{\theta}) + \frac{\mu_1}{2}\left\|\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0) - \boldsymbol{h}\right\|_2^2 \\
& + \frac{\mu_2}{2}\left\|\mathbf{1}_u + (\boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0) - \boldsymbol{a}\right\|_2^2 \\
& + \frac{\mu_3}{2}\left\|\boldsymbol{D}^*\boldsymbol{\alpha}^* - \boldsymbol{t}\right\|_2^2 \\
& + \frac{\mu_4}{2}\left\|\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0 - \boldsymbol{b}\right\|_2^2,
\end{aligned} \tag{14}$$

where the augmented Lagrangian terms are introduced to make the sub-optimizations "more convex". According to [6], the values of non-negative hyper-parameters μ_1, μ_2, μ_3 and μ_4 only affect the times of iteration for ADMM to converge but will not affect the values of the converged solutions of model parameters.

Finally, we solve the following five sub-optimizations to update the values of model parameters, $\boldsymbol{\alpha}^*$, β_0, and slack variables, $\boldsymbol{h}$, $\boldsymbol{a}$, $\boldsymbol{t}$ and $\boldsymbol{b}$, iteratively:

$$\begin{aligned}(\boldsymbol{\alpha}^{*(k+1)},\beta_0^{(k+1)}) &= \arg\min_{\boldsymbol{\alpha}^*,\beta_0} \mathcal{L}_a(\boldsymbol{\alpha}^*,\beta_0,\boldsymbol{h}^{(k)},\boldsymbol{a}^{(k)},\boldsymbol{t}^{(k)},\boldsymbol{b}^{(k)},\boldsymbol{u}_{\boldsymbol{h}}^{(k)},\boldsymbol{u}_{\boldsymbol{a}}^{(k)},\boldsymbol{v}^{(k)},\boldsymbol{q}^{(k)}),\\ \boldsymbol{h}^{(k+1)} &= \arg\min_{\boldsymbol{h}} \mathcal{L}_a(\boldsymbol{\alpha}^{*(k+1)},\beta_0^{(k+1)},\boldsymbol{h},\boldsymbol{u}_{\boldsymbol{h}}^{(k)}),\\ \boldsymbol{a}^{(k+1)} &= \arg\min_{\boldsymbol{a}} \mathcal{L}_a(\boldsymbol{\alpha}^{*(k+1)},\beta_0^{(k+1)},\boldsymbol{a},\boldsymbol{u}_{\boldsymbol{a}}^{(k)}),\\ \boldsymbol{t}^{(k+1)} &= \arg\min_{\boldsymbol{t}} \mathcal{L}_a(\boldsymbol{\alpha}^{*(k+1)},\beta_0^{(k+1)},\boldsymbol{t},\boldsymbol{v}^{(k)}),\\ \boldsymbol{b}^{(k+1)} &= \arg\min_{\boldsymbol{b}} \mathcal{L}_a(\boldsymbol{\alpha}^{*(k+1)},\beta_0^{(k+1)},\boldsymbol{b},\boldsymbol{q}^{(k)}).\end{aligned} \tag{15}$$

Dual variables $\boldsymbol{u}_h$, $\boldsymbol{u}_a$, $\boldsymbol{v}$, $\boldsymbol{q}$ are updated subsequently in the same round of iteration:

$$\begin{aligned}&\boldsymbol{u}_{\boldsymbol{h}}^{(k+1)} = \boldsymbol{u}_{\boldsymbol{h}}^{(k)} + \mu_1[\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_p\beta_0^{(k+1)}) - \boldsymbol{h}^{(k+1)}],\\ &\boldsymbol{u}_{\boldsymbol{a}}^{(k+1)} = \boldsymbol{u}_{\boldsymbol{a}}^{(k)} + \mu_2(\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_u\beta_0^{(k+1)} - \boldsymbol{a}^{(k+1)}).\\ &\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)} + \mu_3(\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} - \boldsymbol{t}^{(k+1)}),\\ &\boldsymbol{q}^{(k+1)} = \boldsymbol{q}^{(k)} + \mu_4(\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_{pu}\beta_0^{(k+1)} - \boldsymbol{b}^{(k+1)}).\end{aligned} \tag{16}$$

3.2.2. *Closed-form solutions to the five sub-optimizations*

In this section, we derive the closed forms of the five sub-optimizations, i.e., the update of $(\boldsymbol{\alpha}^*,\beta_0)$, $\boldsymbol{h}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{t}$.

1. Update of $\boldsymbol{\alpha}^*$ and β_0 The update of $\boldsymbol{\alpha}^*$ and β_0 is

$$\begin{aligned}&\arg\min_{\boldsymbol{\alpha}^*,\beta_0} \frac{\lambda_2}{2}\boldsymbol{\alpha}^{*T}\boldsymbol{G}\boldsymbol{\alpha}^* + {\boldsymbol{u}_{\boldsymbol{h}}^{(k)}}^T[\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0) - \boldsymbol{h}^{(k)}]\\ &+ {\boldsymbol{u}_{\boldsymbol{a}}^{(k)}}^T(\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0 - \boldsymbol{a}^{(k)}) + {\boldsymbol{v}^{(k)}}^T(\boldsymbol{\alpha}^* - \boldsymbol{t}^{(k)})\\ &+ {\boldsymbol{q}^{(k)}}^T(\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0 - \boldsymbol{b}^{(k)}) + \frac{\mu_1}{2}\left\|\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^* + \mathbf{1}_p\beta_0) - \boldsymbol{h}^{(k)}\right\|_2^2\\ &+ \frac{\mu_2}{2}\left\|\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^* + \mathbf{1}_u\beta_0 - \boldsymbol{a}^{(k)}\right\|_2^2 + \frac{\mu_3}{2}\left\|\boldsymbol{\alpha}^* - \boldsymbol{t}^{(k)}\right\|_2^2\\ &+ \frac{\mu_4}{2}\left\|\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^* + \mathbf{1}_{pu}\beta_0 - \boldsymbol{b}^{(k)}\right\|_2^2,\end{aligned} \tag{17}$$

which is a quadratic optimization with every term differentiable. Thus the update of the model parameters $\boldsymbol{\alpha}^*$, β_0 can be obtained by computing

$$\begin{aligned}({\boldsymbol{\alpha}^{*(k+1)}}^T,\beta_0^{(k+1)})^T =& \Big[\mu_3\boldsymbol{D}^{*[0]T}\boldsymbol{D}^{*[0]} + \lambda_2\boldsymbol{G}^{[0]} + \mu_1\boldsymbol{S}_{[p]}^{*T}\boldsymbol{S}_{[p]}^*\\ &+ \mu_2\boldsymbol{S}_{[u]}^{*T}\boldsymbol{S}_{[u]}^* + \mu_4\boldsymbol{S}_{[pu]}^{*T}\boldsymbol{S}_{[pu]}^*\Big]^{-1}\\ &\Big[\boldsymbol{S}_{[p]}^{*T}{\boldsymbol{u}_{\boldsymbol{h}}}^{(k)} - \boldsymbol{S}_{[u]}^{*T}{\boldsymbol{u}_{\boldsymbol{a}}}^{(k)} - \boldsymbol{D}^{*[0]T}\boldsymbol{v}^{(k)[0]}\\ &+\mu_1\boldsymbol{S}_{[p]}^{*T}(\mathbf{1}_p - \boldsymbol{h}^{(k)}) - \mu_2\boldsymbol{S}_{[u]}^{*T}(\mathbf{1}_u - \boldsymbol{a}^{(k)})\\ &+\mu_3\boldsymbol{D}^{*[0]T}\boldsymbol{t}^{(k)[0]} + \mu_4\boldsymbol{S}_{[pu]}^{*T}\boldsymbol{b}^{(k)} - \boldsymbol{S}_{[pu]}^{*T}\boldsymbol{q}^{(k)}\Big],\end{aligned} \tag{18}$$

where

$$\boldsymbol{S}_{[j]}^* = [\boldsymbol{S}_{[j]},\mathbf{1}_j], j=p,u,\quad \boldsymbol{D}^{*[0]} = \begin{bmatrix}\boldsymbol{D}^* & 0\\ 0 & 0\end{bmatrix}.$$

2. Update of $\boldsymbol{h}$ The update of $\boldsymbol{h}$ is

$$\begin{aligned}\boldsymbol{h}^{(k+1)} =& \arg\min_{\boldsymbol{h}} C_p\mathbf{1}_p^T[\boldsymbol{h}]_+ + {\boldsymbol{u}_{\boldsymbol{h}}^{(k)}}^T[\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_p\beta_0^{(k+1)}) - \boldsymbol{h}]\\ &+\frac{\mu_1}{2}\left\|\mathbf{1}_p - (\boldsymbol{S}_{[p]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_p\beta_0^{(k+1)}) - \boldsymbol{h}\right\|_2^2.\end{aligned} \tag{19}$$

Considering the following threshold function,

$$s_c(d) = \arg\min_x c[x]_+ + \frac{1}{2}\|x-d\|_2^2 = \begin{cases} d-c, d > c, \\ 0, 0 \le d \le c, \\ d, d < 0, \end{cases} \tag{20}$$

we can solve the ith element of $\boldsymbol{h}^{(k+1)}$ as

$$h_i^{(k+1)} = s_{\frac{C_p}{\mu_1}}\left[1 + \frac{u_{\boldsymbol{h}i}^{(k)}}{\mu_1} - (\boldsymbol{s}_i^T\boldsymbol{\alpha}^{*(k+1)} + \beta_0^{(k+1)})\right]. \tag{21}$$

3. Update of $\boldsymbol{a}$ The update of $\boldsymbol{a}$

$$\begin{aligned}\boldsymbol{a}^{(k+1)} = \arg\min_{\boldsymbol{a}} C_u\boldsymbol{a}^T\boldsymbol{a} + \boldsymbol{u}_{\boldsymbol{a}}^{(k)^T}[\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_u\beta_0^{(k+1)} - \boldsymbol{a}] \\ + \frac{\mu_2}{2}\left\|(\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_u\beta_0^{(k+1)} - \boldsymbol{a})\right\|_2^2,\end{aligned} \tag{22}$$

which is also a quadratic problem, similar to that for the update of $\boldsymbol{\alpha}^*$ and β_0. Thus we can find the following solution:

$$\boldsymbol{a}^{(k+1)} = \frac{1}{2C_u + \mu_2}[\boldsymbol{u}_{\boldsymbol{a}}^{(k)} + \mu_2(\mathbf{1}_u + \boldsymbol{S}_{[u]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_u\beta_0^{(k+1)})]. \tag{23}$$

4. Update of $\boldsymbol{b}$ The update of $\boldsymbol{b}$ is

$$\begin{aligned}\boldsymbol{b}^{(k+1)} = \arg\min_{\boldsymbol{b}} \boldsymbol{b}^T\boldsymbol{R}\boldsymbol{b} + \boldsymbol{q}^{(k)^T}(\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_{pu}\beta_0^{(k+1)} - \boldsymbol{b}) \\ + \frac{\mu_4}{2}\left\|\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_{pu}\beta_0^{(k+1)} - \boldsymbol{b}\right\|_2^2,\end{aligned} \tag{24}$$

which is also a quadratic problem. Thus $\boldsymbol{b}^{(k+1)}$ can be obtained as

$$\boldsymbol{b}^{(k+1)} = (2\boldsymbol{R} + \mu_4\boldsymbol{I}_{n_p+n_u})^{-1}\left[\boldsymbol{q}^{(k)} + \mu_4(\boldsymbol{S}_{[pu]}\boldsymbol{\alpha}^{*(k+1)} + \mathbf{1}_{pu}\beta_0^{(k+1)})\right]. \tag{25}$$

5. Update of $\boldsymbol{t}$ The update of $\boldsymbol{t}$ is

$$\begin{aligned}\boldsymbol{t}^{(k+1)} &= \arg\min_{\boldsymbol{t}} \frac{\lambda_1}{2}\sum_{i=1}^m \|\boldsymbol{t}_{[i]}\|_2 + \boldsymbol{v}^{(k)^T}(\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} - \boldsymbol{t}) + \frac{\mu_3}{2}\left\|\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} - \boldsymbol{t}\right\|_2^2, \\ &= \arg\min_{\boldsymbol{t}} \frac{\lambda_1}{2}\sum_{i=1}^m \|\boldsymbol{t}_{[i]}\|_2 + \frac{\mu_3}{2}\left\|\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} + \frac{\boldsymbol{v}^{(k)}}{\mu_3} - \boldsymbol{t}\right\|_2^2,\end{aligned} \tag{26}$$

which can be treated as a special case of the following group-lasso regression:

$$\arg\min_{\boldsymbol{t}} \frac{\lambda_1}{2\mu_3}\sum_{i=1}^m \|\boldsymbol{t}_{[i]}\|_2 + \frac{1}{2}\|\boldsymbol{y}_* - \boldsymbol{X}_*\boldsymbol{t}\|_2^2, \tag{27}$$

where $\boldsymbol{y}_* = \boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} + \frac{\boldsymbol{v}^{(k)}}{\mu_3}$ and $\boldsymbol{X}_* = \boldsymbol{I}_{m(m+1)}$.

Then, based on the techniques for the optimization of group variables in [35], we can find the Karush–Kuhn–Tucker (KKT) conditions for $\boldsymbol{t}_{[i]}$ in Equation (27):

$$\begin{aligned} -\boldsymbol{X}_{[i]}^{*T}(\boldsymbol{y}_* - \boldsymbol{X}_*\boldsymbol{t}) + \frac{\lambda_1\boldsymbol{t}_{[i]}}{2\mu_3\left\|\boldsymbol{t}_{[i]}\right\|_2} = \mathbf{0} \quad &\text{if } \boldsymbol{t}_{[i]} \neq \mathbf{0}, \\ \left\|-\boldsymbol{X}_{[i]}^{*T}(\boldsymbol{y}_* - \boldsymbol{X}_*\boldsymbol{t})\right\|_2 \leqslant \frac{\lambda_1}{2\mu_3} \quad &\text{if } \boldsymbol{t}_{[i]} = \mathbf{0}, \end{aligned} \tag{28}$$

where $\boldsymbol{X}_{[i]}^* = \boldsymbol{X}_*[\cdot, (i-1)(m+1)+1 : i(m+1)]$. When $\boldsymbol{X}_* = \boldsymbol{I}_{m(m+1)}$, we can find $\boldsymbol{X}_{[i]}^{*T}\boldsymbol{X}_{[i]}^* = \boldsymbol{I}_{m+1}$.

Therefore, by denoting $\mathbb{S}_i = \boldsymbol{X}_{[i]}^{*T}\left(\boldsymbol{y}_* - \boldsymbol{X}_{[-i]}^*\boldsymbol{t}_{[-i]}\right) = \boldsymbol{X}_{[i]}^{*T}\boldsymbol{y}_*$, it follows Equation (28) that

$$\boldsymbol{t}_{[i]} = \left[1 - \frac{\lambda_1}{2\mu_3\|\mathbb{S}_i\|_2}\right]_+ \mathbb{S}_i, \tag{29}$$

and thus the closed form of the updated $\boldsymbol{t}_{[i]}$ is

$$\boldsymbol{t}_{[i]}^{(k+1)} = \left[1 - \frac{\lambda_1}{2\mu_3\left\|\boldsymbol{X}_{[i]}^{*T}(\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} + \frac{\boldsymbol{v}^{(k)}}{\mu_3})\right\|_2}\right]_+ \boldsymbol{X}_{[i]}^{*T}(\boldsymbol{D}^*\boldsymbol{\alpha}^{*(k+1)} + \frac{\boldsymbol{v}^{(k)}}{\mu_3}). \tag{30}$$

3.2.3. Stopping criteria of the five-block ADMM

According to [3], there are dual stopping criteria to verify the convergence of ADMM. In our case, the dual stopping criteria can be formulated as

$$\begin{aligned}
&\left\|\mu_1[\boldsymbol{S}_{[p]},\mathbf{1}_p]^T(\boldsymbol{h}^{(k)}-\boldsymbol{h}^{(k-1)})\right\|_2 \le \epsilon^{\text{abs}}\sqrt{\frac{m^2+3m}{2}+1}+\epsilon^{\text{rel}}\left\|[\boldsymbol{S}_{[p]},\mathbf{1}_p]^T\boldsymbol{u}_h^{(k)}\right\|_2,\\
&\left\|\mu_2[\boldsymbol{S}_{[u]},\mathbf{1}_u]^T(\boldsymbol{a}^{(k)}-\boldsymbol{a}^{(k-1)})\right\|_2 \le \epsilon^{\text{abs}}\sqrt{\frac{m^2+3m}{2}+1}+\epsilon^{\text{rel}}\left\|[\boldsymbol{S}_{[u]},\mathbf{1}_u]^T\boldsymbol{u}_a^{(k)}\right\|_2,\\
&\left\|\mu_3\boldsymbol{D}^{*T}(\boldsymbol{t}^{(k)}-\boldsymbol{t}^{(k-1)})\right\|_2 \le \epsilon^{\text{abs}}\sqrt{\frac{m^2+3m}{2}+1}+\epsilon^{\text{rel}}\left\|\boldsymbol{D}^{*T}\boldsymbol{v}^{(k)}\right\|_2,\\
&\left\|\mu_4[\boldsymbol{S}_{[pu]},\mathbf{1}_{pu}]^T(\boldsymbol{b}^{(k)}-\boldsymbol{b}^{(k-1)})\right\|_2 \le \epsilon^{\text{abs}}\sqrt{\frac{m^2+3m}{2}+1}+\epsilon^{\text{rel}}\left\|[\boldsymbol{S}_{[pu]},\mathbf{1}_{pu}]^T\boldsymbol{q}^{(k)}\right\|_2,
\end{aligned} \tag{31}$$

where hyper-parameter ϵ^{rel} denotes the relative tolerance of convergence and hyper-parameter ϵ^{abs} denotes the absolute tolerance of convergence.

Finally, we summarise the algorithm of GKF-PUAL with quadratic decision boundary for PU classification in Algorithm 1.

Algorithm 1 GKF-PUAL.

```
Input: PU dataset, $C_p$, $C_u$, $\lambda_1$, $\lambda_2$, $\sigma$ and $\mu_1$, $\mu_2$, $\mu_3$, $\mu_4$
Output: $\boldsymbol{A}$, $\boldsymbol{\beta}$ and $\beta_0$
1: Initialise $\boldsymbol{A}$ to be a zero matrix, $\boldsymbol{\beta}$ to be a vector of ones, and $\beta_0$ to be 1
2: Initialise $\boldsymbol{h}$, $\boldsymbol{a}$, $\boldsymbol{t}$, $\boldsymbol{b}$ via the constraints in Equation (12)
3: Initialise $\boldsymbol{u}_h$, $\boldsymbol{u}_a$, $\boldsymbol{v}$ and $\boldsymbol{q}$ to be vectors of ones
4: Convert $\boldsymbol{A}$ and $\boldsymbol{\beta}$ to $\boldsymbol{\alpha}^*$ following the steps in Section 3.1.2
5: while not fulfil the stopping criteria in (31) do
6:     Update $\boldsymbol{\alpha}^{*(k+1)}$ and $\beta_0^{(k+1)}$ via Equation (18)
7:     Update $\boldsymbol{h}^{(k+1)}$ via Equation (21)
8:     Update $\boldsymbol{a}^{(k+1)}$ via Equation (23)
9:     Update $\boldsymbol{b}^{(k+1)}$ via Equation (25)
10:    Update $\boldsymbol{t}^{(k+1)}$ via Equation (30)
11:    Update $\boldsymbol{u}_h^{(k+1)}$, $\boldsymbol{u}_a^{(k+1)}$, $\boldsymbol{v}^{(k+1)}$ and $\boldsymbol{q}^{(k+1)}$ via Equation (16)
12: end while
13: Convert $\boldsymbol{\alpha}^*$ back to $\boldsymbol{A}$ and $\boldsymbol{\beta}$.
```

4. Experiments

In this section, experiments on real datasets were conducted to compare our proposed GKF-PUAL with PUAL and other established PU, classifiers, i.e., GLLC, uPU and nnPU.

4.1. Datasets and preprocessing

For the evaluation of PU classifiers, researchers usually generate PU datasets from publicly-available benchmark datasets where both positive and negative labels are available [5,14,20]. Following this strategy, we choose 13 UCI datasets of various types of real-world data, such as handwritten digit images, clinical records and ecoli bacteria. Specifically, the datasets chosen in this study are Accelerometer (**Acc**), **Ecoli**, Pen-based recognition of handwritten digits (**Pen**), Online retail (**OR1**), Online retail II (**OR2**), Sepsis survival minimal clinical records (**SSMCR**), Raisin Dataset (**RD**), Occupancy detection (**OD**), User knowledge modelling (**UMD**), **Seeds**, Energy efficiency (**ENB**), Liver disorders (**LD**) and Heart disease (**HD**). The details of these 13 datasets are summarised in Table 1.

To verify both the classification and variable selection performance, the 13 datasets were preprocessed through the following two steps:

1. Standardise the 13 real-world datasets in Table 1.
2. Add irrelevant variables into the standardised dataset. Each of these irrelevant variables follows standard normal distribution, and the number of these irrelevant variables is twice the number of the original variables.

The standardisation in Step 1 is to ensure that the introduced irrelevant variables in Step 2 exert sufficient disturbance on the original datasets for model training.

4.2. Training-test split

The training-test split for the 13 real-world datasets was done through the steps:

Table 1
Summary of the datasets used in experiments.

Dataset	positive instances	negative instances	#variables
Acc	100 red	100 blue	4
Ecoli	116 im & 52 pp	143 cp & 25 om	6
Pen	200 one & 200 eight	400 four versicolor	16
OR1	301 UK	301 Germany	4
OR2	500 UK	500 Germany	4
SSMCR	391 alive	109 dead	3
OD	100 occupied	300 not occupied	5
UMD	83 Low	63 high	5
Seeds	70 Kama	70 Rosa	7
ENB	144 TypeII	144 Type III	7
RD	450 Kecimen	450 Besni	7
LD	144 class 1	200 class 2	6
HD	150 absence	119 presence	13

1. γ' of the positive instances in the real-world dataset were selected randomly into the labelled-positive set, and the rest of the positive instances were put into the unlabeled set.
2. The whole labelled-positive set and 70% of the unlabeled set formed the training set. The rest 30% of the unlabeled set formed the test set.

Let label frequency γ denote the proportion of labelled positive instances among all the positive instances in the training set. In this case, we obtained 10 pairs of PU training and test sets for each of the 13 datasets with a certain label frequency $\gamma = 0.5$ or 0.25 in the case-control scenario [1], where the unlabeled set and the test set can be regarded as i.i.d. samples from the same population. We chose two challenging situations with relatively low label frequencies, i.e. $\gamma = 0.5$ or 0.25. To achieve these two label frequencies, γ' is set to $\frac{7}{17}$ and $\frac{7}{37}$, respectively, given $\gamma = \gamma'/(0.3\gamma' + 0.7)$.

4.3. *Compared methods and model setting*

PUAL, GLLC, uPU and nnPU were trained on the 13 real-world datasets as the compared methods with GKF-PUAL. Furthermore, to demonstrate the effectiveness of variable selection by group lasso, we also test the classification performance of the kernel-free quadratic PUAL (KF-PUAL) without group lasso, using the objective function in Equation (7).

It should be noted that F1-score needs the information of ground-truth labels to be computed, which is not available for the negative instances during the training of PU classifiers. In this case, PUF-score [23] was chosen as the metric for hyper-parameter tuning, which holds a similar pattern to F1-score and does not need the information of ground-truth labels of negative instances.

By fixing C_p to 1 and the number K of the nearest neighbors to 5, λ_1, λ_2 and σ were firstly tuned by 4-fold CV, according to average PUF-score

$$\frac{\text{recall}^2}{P[\text{sgn}(f(\boldsymbol{x}))=1]},$$

where 'recall' can be estimated by $\frac{1}{n_p}\sum_{\boldsymbol{x}_i\in p}\mathbb{I}(\text{sgn}(f(\boldsymbol{x}_i))=1)$ with $\mathbb{I}(\cdot)$ an indicator function and $P[\text{sgn}(f(\boldsymbol{x}))=1]$ can be estimated by $\frac{1}{n_u}\sum_{\boldsymbol{x}_i\in u}\mathbb{I}(\text{sgn}(f(\boldsymbol{x}_i))=1)$, since the unlabeled set can be regarded as i.i.d. samples from the population in the case-control scenario. Specifically, λ_1, λ_2 and σ were tuned from the set $\{10^{-4},10^{-3},\dots,10^{3},10^{4}\}$; C_u was tuned from the set $\{0.5, 0.3, 0.1, 0.05, 0.01\}$. Then λ_1, λ_2 and C_u were continually tuned by following a greedy algorithm as

1. Set λ_1, λ_2, σ and C_u to the best combination from the grid search.
2. Sequentially update one of hyper-parameters λ_1, λ_2, σ and C_u by increasing/decreasing 10% of its current value with the rest of the hyper-parameters unchanged. The optimal scenario on 4-fold CV is set to be the final update of this step.
3. Repeat Step 2 until there is no better scenario appearing.

As mentioned in Section 3.2.1, the positive hyper-parameters μ_1, μ_2, μ_3 and μ_4 will not affect the value of the converged solution of model parameters, hence we fixed μ_1, μ_2, μ_3 and μ_4 to 0.01. In addition, the hyper-parameters ϵ^{rel} and ϵ^{abs} for the stopping criteria in Section 3.2.3 were set to 0.001 and 0.25 respectively by following [3].

The hyper-parameter tuning of PUAL and GLLC follows the same steps after we substitute λ for λ_1 and λ_2. The recommended set in [22] was used as the hyper-parameters of uPU and nnPU as at https://github.com/kiryor/nnPUlearning. Finally, the Radial Basis Function (RBF) kernel was applied to PUAL and GLLC for non-linear decision boundaries.

Table 2
The average F1-score (%) with the standard deviation of the classifiers trained on the 13 real-world PU datasets; for each dataset, the two rows were obtained under label frequencies $\gamma = 0.5$ and 0.25, respectively; the best result is in blue.

Dataset	GKF-PUAL	KF-PUAL	PUAL	GLLC	uPU	nnPU
OD	100.0 ± 0.0	92.54 ± 1.74	85.7 ± 6.1	100.0 ± 0.0	80.0 ± 42.2	100.0 ± 0.0
	91.4 ± 12.3	84.24 ± 1.88	80.6 ± 7.9	100.0 ± 0.0	70.0 ± 48.3	100.0 ± 0.0
OR1	88.3 ± 3.5	75.57 ± 2.89	78.4 ± 2.4	74.3 ± 4.3	15.8 ± 31.6	78.5 ± 6.5
	79.3 ± 12.3	72.66 ± 8.30	73.7 ± 3.4	70.6 ± 6.0	19.6 ± 31.3	67.6 ± 6.6
OR2	80.3 ± 8.2	71.09 ± 2.66	73.4 ± 1.8	71.6 ± 5.9	71.8 ± 6.2	76.9 ± 4.2
	80.7 ± 7.1	70.96 ± 5.33	70.2 ± 6.8	67.4 ± 5.8	69.3 ± 4.5	72.7 ± 3.8
UMD	91.3 ± 3.1	78.24 ± 9.80	74.4 ± 4.1	85.3 ± 7.4	100.0 ± 0.0	100.0 ± 0.0
	90.0 ± 4.6	72.14 ± 6.81	72.4 ± 4.6	84.5 ± 4.3	100.0 ± 0.0	100.0 ± 0.0
Acc	71.2 ± 5.3	64.35 ± 2.36	70.0 ± 4.9	69.8 ± 5.1	19.2 ± 26.4	19.7 ± 27.6
	71.6 ± 5.4	64.89 ± 1.79	69.8 ± 5.1	57.0 ± 8.3	21.3 ± 28.7	22.6 ± 30.3
Ecoli	89.0 ± 0.9	82.25 ± 2.30	91.6 ± 3.6	89.0 ± 3.5	76.1 ± 5.6	77.7 ± 6.1
	85.4 ± 7.2	80.42 ± 4.84	83.4 ± 5.7	86.0 ± 3.9	76.1 ± 6.4	77.7 ± 5.9
ENB	66.6 ± 5.3	47.94 ± 1.10	45.3 ± 7.4	44.1 ± 4.7	28.9 ± 20.2	29.3 ± 21.7
	64.4 ± 6.5	44.21 ± 2.88	44.6 ± 9.4	35.1 ± 6.2	24.5 ± 28.5	24.9 ± 29.1
HD	75.3 ± 4.3	74.68 ± 3.67	80.5 ± 3.1	81.7 ± 5.3	67.5 ± 4.1	70.6 ± 2.1
	76.3 ± 3.0	74.64 ± 2.13	81.2 ± 5.1	80.1 ± 5.2	66.7 ± 4.0	68.9 ± 2.8
Pen	88.3 ± 10.8	76.38 ± 12.45	82.5 ± 21.2	79.0 ± 12.1	83.7 ± 9.4	84.5 ± 12.9
	85.8 ± 12.5	75.95 ± 14.37	80.8 ± 23.9	77.9 ± 9.0	79.8 ± 10.7	80.5 ± 12.3
LD	58.2 ± 5.0	44.78 ± 2.93	47.3 ± 6.3	51.5 ± 6.8	10.9 ± 23.5	29.1 ± 25.6
	58.8 ± 4.1	39.45 ± 5.45	45.4 ± 5.3	50.8 ± 3.9	9.5 ± 21.7	18.8 ± 24.6
SSMCR	88.2 ± 1.3	80.48 ± 5.92	82.8 ± 2.8	89.2 ± 1.7	79.5 ± 1.5	80.7 ± 1.3
	87.9 ± 1.9	74.09 ± 2.75	82.3 ± 2.2	88.6 ± 1.8	79.6 ± 1.9	81.9 ± 1.4
Seeds	86.3 ± 4.2	78.15 ± 4.01	74.2 ± 7.2	75.4 ± 11.3	84.1 ± 4.0	91.4 ± 5.4
	89.9 ± 3.4	73.80 ± 1.71	70.8 ± 21.2	74.9 ± 13.6	81.0 ± 5.9	85.6 ± 5.2
RD	32.1 ± 2.2	24.64 ± 4.05	77.1 ± 2.9	78.1 ± 4.8	68.6 ± 14.7	69.9 ± 16.3
	31.8 ± 2.6	23.82 ± 7.39	71.8 ± 4.3	75.1 ± 5.8	67.0 ± 14.1	69.5 ± 12.7

4.4. Results and analysis

4.4.1. Classification performance

The results of the experiments are summarised by average F1-score in Table 2, from which we can make the following observations.

Firstly, GKF-PUAL outperformed PUAL on 21 out of a total of 26 cases. This supports our motivation that GKF-PUAL can achieve better PU classification performance than PUAL on datasets containing irrelevant variables. Moreover, GKF-PUAL consistently outperformed KF-PUAL, indicating the effectiveness of variable selection through group lasso. It is also worth noting that KF-PUAL only surpassed PUAL in 7 out of 26 cases, suggesting that quadratic decision boundary lacks flexibility. However, when variable selection was enabled, GKF-PUAL demonstrated dominant classification performance against PUAL. Secondly, there are 14 out of 26 cases where GKF-PUAL performs the best among all the five methods compared in the experiments. This indicates superior classification performance of GKF-PUAL. Thirdly, on all 6 cases of the 3 trifurcate PU datasets, **OR1**, **OR2** and **Pen**, GKF-PUAL performs the best. This verifies that GKF-PUAL not only preserves the superior classification performance of PUAL addressing trifurcate data, but also does so without being affected by a large number of irrelevant variables.

4.4.2. Variable selection performance

The variable-selection performance of GKF-PUAL for each dataset is summarised in Table 3. The kernel trick adopted by PUAL made it impossible to observe the coefficients of individual variables in the model trained by PUAL. Hence Table 3 only summarises for GKF-PUAL, from which we can observe the following patterns.

Firstly, generally GKF-PUAL can remove a large proportion of irrelevant variables from most datasets. Secondly, on two datasets **HD** and **RD**, the proportion of irrelevant variables removed by GKF-PUAL is lower than 25% on average. As GKF-PUAL does not perform well as PUAL on these two datasets, especially on **RD**, a potential reason for this under-performed variable selection is that the quadratic boundary generated in the kernel-free setting in Equation (11) is not sufficiently complex for the data distribution of **HD** and **RD**. This can be verified by the t-SNE plots of the two datasets in Fig. 3. Clearly, the two classes are heavily mixed together, indicating the need for a more flexible decision boundary.

Table 3
The average percentage (with the standard deviation) of the irrelevant variables whose coefficients were compressed to zero relative to the total number of irrelevant variables. For each dataset, the two rows are for $\gamma = 0.25$ and 0.5, respectively.

OD	84.00 ± 6.99 86.00 ± 6.99	SSMCR	55.00 ± 15.81 53.33 ± 20.49
OR1	75.00 ± 11.79 73.75 ± 10.94	UMD	86.00 ± 9.66 76.00 ± 23.66
OR2	52.50 ± 11.49 46.25 ± 13.76	RD	24.29 ± 9.04 15.71 ± 8.11
Pen	70.94 ± 8.21 67.19 ± 6.95	Seeds	47.14 ± 9.04 51.43 ± 11.07
Ecoli	77.08 ± 11.85 72.92 ± 12.87	HD	17.69 ± 4.87 15.00 ± 6.14
ENB	48.57 ± 7.38 47.86 ± 9.55	LD	56.67 ± 11.65 55.83 ± 11.15
Acc	91.25 ± 9.22 85.00 ± 6.45		

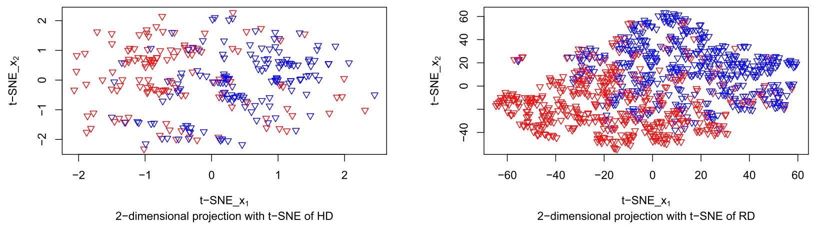

Fig. 3. The t-SNE plots of **HD** and **RD**. The rest of the caption is as in Fig. 1.

5. Conclusion

In this paper, we propose GKF-PUAL, a kernel-free quadratic PU classifier with the group-lasso regularisation for embedded variable selection. The key novelty of GKF-PUAL is to integrate the idea of PUAL to introduce an asymmetric structure of loss on positive instances, the idea of SQS-SVM to generate quadratic decision boundary without using the kernel trick, and the idea of group lasso to achieve embedded variable selection. The superior classification and variable selection performance of GKF-PUAL is demonstrated by experiments on real-world datasets.

We identify two areas for improvement in future work. First, the quadratic decision boundary limits GKF-PUAL's capacity to classify the data that are beyond quadratically separable. Second, as indicated in Table 3, the group-lasso regularisation still does not remove irrelevant features completely. Hence, a more flexible kernel-free method with stronger variable-selection capacity needs to be investigated as our future work.

CRediT authorship contribution statement

Xiaoke Wang: Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Rui Zhu:** Writing – review & editing, Supervision. **Jing-Hao Xue:** Writing – review & editing, Supervision.

Declaration of competing interest

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

Data availability

Data are publicly available.

References

[1] J. Bekker, J. Davis, Learning from positive and unlabeled data: a survey, Mach. Learn. 109 (4) (2020) 719–760.

[2] V. Bolón-Canedo, N. Sánchez-Marono, A. Alonso-Betanzos, J.M. Benítez, F. Herrera, A review of microarray datasets and applied feature selection methods, Inf. Sci. 282 (2014) 111–135.
[3] S. Boyd, N. Parikh, E. Chu, Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers, Now Publishers Inc, 2011.
[4] G. Buch, A. Schulz, I. Schmidtmann, K. Strauch, P.S. Wild, A systematic review and evaluation of statistical methods for group variable selection, Stat. Med. 42 (3) (2023) 331–352.
[5] B. Calvo, P. Larranaga, J.A. Lozano, Feature subset selection from positive and unlabelled examples, Pattern Recognit. Lett. 30 (11) (2009) 1027–1036.
[6] C. Chen, B. He, Y. Ye, X. Yuan, The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent, Math. Program. 155 (1) (2016) 57–79.
[7] X. Chen, C. Gong, J. Yang, Cost-sensitive positive and unlabeled learning, Inf. Sci. 558 (2021) 229–245.
[8] F. Chiaroni, G. Khodabandelou, M.-C. Rahal, N. Hueber, F. Dufaux, Counter-examples generation from a positive unlabeled image dataset, Pattern Recognit. 107 (2020) 107527.
[9] L.M. de Campos, J.M. Fernández-Luna, J.F. Huete, L. Redondo-Expósito, Positive unlabeled learning for building recommender systems in a parliamentary setting, Inf. Sci. 433 (2018) 221–232.
[10] L. de Carvalho Pagliosa, R.F. de Mello, Semi-supervised time series classification on positive and unlabeled problems using cross-recurrence quantification analysis, Pattern Recognit. 80 (2018) 53–63.
[11] M. Du Plessis, G. Niu, M. Sugiyama, Convex formulation for learning from positive and unlabeled data, in: International Conference on Machine Learning, PMLR, 2015, pp. 1386–1394.
[12] M.C. Du Plessis, G. Niu, M. Sugiyama, Analysis of learning from positive and unlabeled data, Adv. Neural Inf. Process. Syst. 27 (2014).
[13] L. Freijeiro-González, M. Febrero-Bande, W. González-Manteiga, A critical review of LASSO and its derivatives for variable selection under dependence among covariates, Int. Stat. Rev. 90 (1) (2022) 118–145.
[14] K. Furmańczyk, K. Paczutkowski, M. Dudziński, D. Dziewa-Dawidczyk, Classification and feature selection methods based on fitting logistic regression to PU data, J. Comput. Sci. 72 (2023) 102095.
[15] M. González, C. Bergmeir, I. Triguero, Y. Rodríguez, J.M. Benítez, On the stopping criteria for k-nearest neighbor in positive unlabeled time series classification problems, Inf. Sci. 328 (2016) 42–59.
[16] F. He, T. Liu, G.I. Webb, D. Tao, Instance-dependent PU learning by Bayesian optimal relabeling, arXiv preprint, arXiv:1808.02180.
[17] Y. He, X. Li, M. Zhang, P. Fournier-Viger, J.Z. Huang, S. Salloum, A novel observation points-based positive-unlabeled learning algorithm, in: CAAI Transactions on Intelligence Technology, 2023.
[18] H.S. Helm, A. Basu, A. Athreya, Y. Park, J.T. Vogelstein, C.E. Priebe, M. Winding, M. Zlatic, A. Cardona, P. Bourke, et al., Distance-based positive and unlabeled learning for ranking, Pattern Recognit. 134 (2023) 109085.
[19] A. Kaboutari, S. Branch, J. Bagherzadeh, F. Kheradmand, An evaluation of feature selection methods for positive-unlabeled learning in text classification, Int. J. Comput. Appl. Technol. Res. 3 (9) (2014) 595–599.
[20] T. Ke, L. Jing, H. Lv, L. Zhang, Y. Hu, Global and local learning from positive and unlabeled examples, Appl. Intell. 48 (8) (2018) 2373–2392.
[21] T. Ke, H. Lv, M. Sun, L. Zhang, A biased least squares support vector machine based on Mahalanobis distance for PU learning, Phys. A, Stat. Mech. Appl. 509 (2018) 422–438.
[22] R. Kiryo, G. Niu, M.C.d. Plessis, M. Sugiyama, Positive-unlabeled learning with non-negative risk estimator, arXiv preprint, arXiv:1703.00593.
[23] W.S. Lee, B. Liu, Learning with positive and unlabeled examples using weighted logistic regression, in: ICML, vol. 3, 2003, pp. 448–455.
[24] B. Liu, Y. Dai, X. Li, W.S. Lee, P.S. Yu, Building text classifiers using positive and unlabeled examples, in: Third IEEE International Conference on Data Mining, IEEE, 2003, pp. 179–186.
[25] B. Liu, J. Liu, Y. Xiao, Q. Chen, K. Wang, R. Huang, L. Li, A new self-paced learning method for privilege-based positive and unlabeled learning, Inf. Sci. 609 (2022) 996–1009.
[26] Z. Liu, W. Shi, D. Li, Q. Qin, Partially supervised classification: based on weighted unlabeled samples support vector machine, in: Data Warehousing and Mining: Concepts, Methodologies, Tools, and Applications, IGI Global, 2008, pp. 1216–1230.
[27] J. Luo, S.-C. Fang, Z. Deng, X. Guo, Soft quadratic surface support vector machine for binary classification, Asia-Pac. J. Oper. Res. 33 (06) (2016) 1650046.
[28] L. Meier, S. Van De Geer, P. Bühlmann, The group lasso for logistic regression, J. R. Stat. Soc., Ser. B, Stat. Methodol. 70 (1) (2008) 53–71.
[29] J.T. Pintas, L.A. Fernandes, A.C.B. Garcia, Feature selection methods for text classification: a systematic literature review, Artif. Intell. Rev. 54 (8) (2021) 6149–6200.
[30] K. Ren, H. Yang, Y. Zhao, W. Chen, M. Xue, H. Miao, S. Huang, J. Liu, A robust AUC maximization framework with simultaneous outlier detection and feature selection for positive-unlabeled classification, IEEE Trans. Neural Netw. Learn. Syst. 30 (10) (2018) 3072–3083.
[31] G. Su, W. Chen, M. Xu, Positive-unlabeled learning from imbalanced data, in: Proceedings of the 30th International Joint Conference on Artificial Intelligence, Virtual Event, 2021.
[32] J. Tang, S. Alelyani, H. Liu, Feature selection for classification: a review, in: Data Classification: Algorithms and Applications, 2014, pp. 37–64.
[33] X. Wang, X. Yang, R. Zhu, J.-H. Xue, PUAL: a classifier on trifurcate positive-unlabeled data, arXiv preprint, arXiv:2405.20970.
[34] C. Xu, C. Liu, S. Yang, Y. Wang, S. Zhang, L. Jia, Y. Fu, Split-PU: hardness-aware training strategy for positive-unlabeled learning, in: Proceedings of the 30th ACM International Conference on Multimedia, 2022, pp. 2719–2729.
[35] M. Yuan, Y. Lin, Model selection and estimation in regression with grouped variables, J. R. Stat. Soc., Ser. B, Stat. Methodol. 68 (1) (2006) 49–67.