Pattern Recognition 169 (2026) 111892

Contents lists available at ScienceDirect

Pattern Recognition

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

UC-PUAL: A universally consistent classifier of positive-unlabelled data

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

[a] Department of Statistical Science, University College London, London, WC1E 6BT, UK
[b] Bayes Business School, City St George's, University of London, London, EC1Y 8TZ, UK

ARTICLE INFO

Keywords:
Positive-unlabelled learning
Universal consistency
Trifurcate data

ABSTRACT

Positive-unlabelled (PU) learning is a challenging task in pattern recognition, as there are only labelled-positive instances and unlabelled instances available for the training of a classifier. The task becomes even harder when the PU data show an underlying trifurcate pattern that positive instances roughly distribute on both sides of ground-truth negative instances. To address this issue, we propose a universally consistent PU classifier with asymmetric loss (UC-PUAL) on positive instances. We also propose two three-block algorithms for non-convex optimisation to enable UC-PUAL to obtain linear and kernel-induced non-linear decision boundaries, respectively. Theoretical and experimental results verify the superiority of UC-PUAL. The code for UC-PUAL is available at https://github.com/tkks22123/UC-PUAL.

1. Introduction

Positive-unlabelled (PU) learning is to build classifiers with only labelled-positive instances and unlabelled instances available for training. In other words, no labelled-negative instances are available for the classifier training. In practice, there is a wide range of real-world applications of PU learning, for example, time series classification [1], learning to rank for recommendation systems [2].

There are mainly two ways to learn a PU classifier. One way can be termed multi-step approach [3–7], which trains a series of classifiers: firstly a classifier to search the unlabelled set for pseudo-negative instances (i.e., the instances with high likelihood to be negative); and then a semi-supervised classifier based on the labelled-positive instances, the pseudo-negative instances, and the unlabelled instances. Generative adversarial networks were also tailored in the first step to generate pseudo-negative instances [8]. The other way can be termed one-step approach [9–12]. Since an inadequate classifier at the first step of multi-step methods can trigger undesired chain reaction and unsatisfactory final performance, it is of risk to apply a multi-step method in practice. This paper focuses on one-step methods.

The one-step methods can be further divided into two types. The first type is inconsistent methods [13–15], whose objective functions are not consistent estimators of the expected loss of classification. They treat all the unlabelled instances as negative during model training. Inconsistent methods include large margin-based approaches such as the biased SVM (BSVM) [13], which assigns lower weights to the unlabelled data; the biased least squares SVM (BLS-SVM) [14], which replaces the hinge loss with the squared loss; and the global and local learning classifier (GLLC) [15], which incorporates local information.

The second type of one-step methods is consistent methods [16–18]. In contrast to the inconsistent methods, the objective functions of consistent methods were crafted to be consistent estimators of the expected loss of classification from the data population. A pioneer consistent method is the unbiased PU learning (uPU) method [16], which treats all unlabelled instances as a mix of positive and negative instances and includes two unbiased and consistent estimators of the expected loss for the unlabelled set.

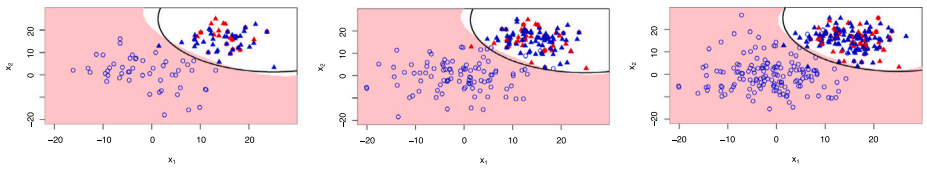
In practice, it often occurs a trifurcate pattern underlying real-world datasets, where positive instances roughly distribute on both sides of ground-truth negative instances [19]. A classifier with non-linear decision boundary will be needed to classify such data, which can be achieved via kernel trick. However, in the kernel-induced new feature space, the two positive subsets can have very distinct distances from the ideal decision boundary. In this case, using a loss function like the squared loss, as does GLLC, will incorrectly impose big penalties on the correctly classified positive instances (i.e., the set of positive instances far away from the ideal decision boundary) and hence drag the decision boundary towards one which misclassifies many more instances. Such an underlying trifurcate pattern is hard to be addressed by the current inconsistent one-step methods like GLLC.

To address this issue, a method called the PU classifier with asymmetric loss (PUAL) was proposed in [19]. PUAL is constructed on the basis of the inconsistent objective method, GLLC [15], but introduces a new structure of asymmetric loss on positive instances: using the hinge loss on the labelled-positive instances, while using the squared loss for the unlabelled instances (thus the unlabelled-positive instances).

Therefore, in this paper, we aim to exploit the best of both worlds: we integrate the ideas from both PUAL, an inconsistent method, and

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

https://doi.org/10.1016/j.patcog.2025.111892
Received 26 June 2024; Received in revised form 30 April 2025; Accepted 21 May 2025
Available online 6 June 2025



(a) Sample size = 100 (b) Sample size = 200 (c) Sample size = 300

Fig. 1. Universal consistency of UC-PUAL. Pink area: negative areas determined by UC-PUAL; Black curve: the decision boundary of the Bayes classifier; Blue circles: unlabelled negative instances; Blue triangles: unlabelled positive instances; Red triangles: labelled-positive instances. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

uPU, a consistent method, to propose a universally consistent PU classifier with asymmetric loss (UC-PUAL) for better classification on trifurcate PU datasets. While uPU ensures consistency with the expected loss of classification, it does not guarantee consistency with the Bayes classifier that achieves the minimum classification error. In contrast, UC-PUAL is designed so that its 0–1 risk for classification is consistent with the Bayes risk when a universal kernel is applied to the objective function. This universally consistent PU classifier ensures that, as the sample size increases, UC-PUAL becomes closer to the theoretically ideal Bayes classifier, as illustrated in the toy examples in Fig. 1.

The novelty and contributions of this paper can be summarised as follows.

Firstly, by integrating the ideas and settings of the objective functions of PUAL and uPU, we propose UC-PUAL, a classifier universally consistent with the Bayes classifier for trifurcate PU data. To the best of our knowledge, this is the first PU classifier specifically designed to achieve this property.

Secondly, we propose two three-block algorithms for non-convex optimisation to enable UC-PUAL to generate linear and kernel-induced non-linear decision boundaries, respectively.

Thirdly, we provide theoretical analysis to show that the universal consistency with the Bayes classifier holds for UC-PUAL.

Finally, we conduct experiments on both synthetic and real-world datasets to showcase the superior performance of UC-PUAL.

2. Related work

In this section, we discuss in more detail about one-step PU classifiers and those closely related to our work.

2.1. Inconsistent methods

For inconsistent methods, a pioneer is BSVM [13], which was proposed on the basis of classic support vector machine, assigning the loss on the unlabelled set a lower weight in the objective function. Then, [14] proposed BLS-SVM by introducing the squared loss, on both labelled-positive set and unlabelled set, into the objective function to avoid the distraction from the unlabelled-positive instances to the model training. Moreover, [15] leveraged local information for the model training, by incorporating a local similarity term into the objective function of BLS-SVM, leading to GLLC. Besides the common frameworks of PU learning, tree methods were leveraged by [20–22] for PU learning.

2.2. Consistent methods

The uPU [16] contains two unbiased and consistent estimators of the expected loss on the unlabelled set by treating all unlabelled instances to be either positive or negative. Noticing that the optimisation of uPU may sometimes fail to converge, nnPU [17] was proposed by introducing a lower threshold to the objective function of uPU, which is a biased but still a consistent estimator of the expected classification loss. Imbalanced nnPU [18] was then proposed by re-weighting the loss in the objective function of nnPU for better classification on imbalanced PU data. [23] modified the non-convex formulation of uPU to a convex version with a novel double hinge loss. A rebalanced version of [23] and an objective function to maximise the expected AUC [24] were proposed also for class imbalanced classification [25]. Pin-LFCS [26] was proposed for robust PU learning via the pinball loss function. PUe [27] and SLPU [28] were proposed for better classification performance on the PU data with selection bias, by incorporating the prior knowledge of the unlabelled data and labelling mechanism into the model training. [29] introduced few-shot learning to handle the label imbalanced PU data. Moreover, [30] generalised the framework of binary PU learning to that of multi-class PU learning via multi-task self-supervised training.

2.3. Detail of three closely related methods

Now we provide more details about three closely related methods, GLLC, PUAL and uPU. Suppose the dataset contains n_p labelled-positive instances and n_u unlabelled instances with m features. Then let $\boldsymbol{X}_{[pu]} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n_p}, \dots, \boldsymbol{x}_{n_p+n_u})^T \in \mathbb{R}^{(n_p+n_u)\times m}$ denote the matrix of features, where vector $\boldsymbol{x}_i \in \mathbb{R}^{m\times 1}$ is the vector of the features of the ith instance; let $\boldsymbol{X}_{[p]} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_{n_p})^T \in \mathbb{R}^{n_p\times m}$ be the feature matrix of the labelled-positive instances; and let $\boldsymbol{X}_{[u]} = (\boldsymbol{x}_{n_p+1}, \dots, \boldsymbol{x}_{n_p+n_u})^T \in \mathbb{R}^{n_u\times m}$ denote the feature matrix of the unlabelled set. Let s be the labelling indicator with $s = 1$ for labelled(-positive) instance.

2.3.1. uPU [16]

Let $l(f(\boldsymbol{X};\boldsymbol{\beta}), Y)$ denote the loss function of the predictive score function f for an instance with its feature $\boldsymbol{X}$ and class indicator $Y \in \{-1, 1\}$ treated as r.v.. The objective function of uPU [16] to be minimised was proposed to be an unbiased and consistent estimator for the expected loss $\mathbb{E}[l(f(\boldsymbol{X};\boldsymbol{\beta}), Y)]$, which can be formulated as

$$\pi \hat{L}_p^1(f) + \hat{L}_u^{-1}(f) - \pi \hat{L}_p^{-1}(f), \tag{1}$$

where $\hat{L}_p^1(f) = \frac{1}{n_p}\sum_{\boldsymbol{x}\in\boldsymbol{X}_{[p]}} l(f(\boldsymbol{x};\boldsymbol{\beta}), 1)$, $\hat{L}_u^{-1}(f) = \frac{1}{n_u}\sum_{\boldsymbol{x}\in\boldsymbol{X}_{[u]}} l(f(\boldsymbol{x};\boldsymbol{\beta}), -1)$, $\hat{L}_p^{-1}(f) = \frac{1}{n_p}\sum_{\boldsymbol{x}\in\boldsymbol{X}_{[p]}} l(f(\boldsymbol{x};\boldsymbol{\beta}), -1)$, and the class prior $\pi = P[Y = 1]$. It should be noted that, although we do not know the ground-truth class y_i for the ith instance, the following objective function of average loss in Eq. (2) is also an unbiased and consistent estimator of the expected loss of classification, $\mathbb{E}[l(f(\boldsymbol{X};\boldsymbol{\beta}), Y)]$, i.e.,

$$\frac{1}{n_u}\sum_{i=1}^{n_u} l(f(\boldsymbol{x}_i;\boldsymbol{\beta},\beta_0), y_i). \tag{2}$$

In this case, the objective function of uPU in Eq. (1) is consistent with Eq. (2) in probability, i.e. $\forall\epsilon$, $P\left[\pi \hat{L}_p^1(f) + \hat{L}_u^{-1}(f) - \pi \hat{L}_p^{-1}(f) - \frac{1}{n_u}\sum_{i=1}^{n_u} l(f(\boldsymbol{x}_i;\boldsymbol{\beta},\beta_0), y_i) < \epsilon\right] \to 1$, with n_p and n_u tending to infinity.

2.3.2. GLLC [15]

As the information of local similarity among instances is helpful for classification, GLLC [15] was proposed by combining BLS-SVM with the local similarity, trained from

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} &+ \frac{c_p}{n_p}[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)] \\ &+ \frac{c_u}{n_u}[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]^T[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)] \\ &+ (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0),\end{aligned} \tag{3}$$

where c_p, c_u, $\lambda > 0$ are hyper-parameters of model, and $\mathbf{1}_{p,u} = (\underbrace{1,1,\ldots,1}_{k})^T, k = n_p, n_u$, and $\boldsymbol{R}$ denotes a local similarity matrix for the instances based on their K-nearest neighbours (See details in [15]).

The predictive score function of GLLC is as simple as

$$f = \boldsymbol{x}^T\boldsymbol{\beta} + \beta_0, \tag{4}$$

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

2.3.3. PUAL [19]

The squared loss in the objective function of GLLC, i.e., Eq. (3), can ensure every instance to contribute to the construction of the decision boundary of BLS-SVM, so that the importance of an unlabelled-positive instance treated as negative can be restricted. However, the squared loss on the label-positive instances will unfortunately impose undesired penalties to the correctly classified positive instances, especially the ones holding long distance from the ideal decision boundary in trifurcate PU data. To address this issue, PUAL [19] was proposed to replace the squared loss by the hinge loss for the labelled-positive instances, while keeping the squared loss for the unlabelled (positive and negative) instances. Therefore, the objective function of PUAL can be formulated as

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{c_p}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ \\ + \frac{c_u}{n_u}[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]^T[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)] \\ + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0),\end{aligned} \tag{5}$$

where $[g(\cdot)]_+$ is a column vector contained by the maximum between each element of $g(\cdot)$ and 0.

We note that both PUAL and the UC-PUAL to be proposed in this paper simply use the same predictive score function as does GLLC in Eq. (4).

The classification performance of PUAL is sensitive to the hyper-parameters c_p and c_u to weigh the losses on the labelled-positive set and the unlabelled set, respectively. However, in UC-PUAL, there is no need to tune these two hyper-parameters.

PUAL does not have a consistent objective function and is not universally consistent to the Bayes classifier. Although uPU has a consistent objective function with respect to the expected loss, it lacks the penalisation on $\boldsymbol{\beta}$ and the use of kernel mapping, therefore does not achieve universal consistency. In the Appendix, we provide further details on determining whether a classifier is universally consistent based on the objective function in Eq. (A.5) with the loss function in Eq. (A.6). While the expectation of the loss function for training may be low, the expected 0-1 loss of uPU and PUAL can still be significantly higher than that of the Bayes classifier. Consequently, using an unsuitable loss function for uPU training can result in poor classification performance, even with a large dataset. However, UC-PUAL is universally consistent to the Bayes classifier and there is no need to select the loss function when the training set is large.

3. Methodology

3.1. UC-PUAL with linear decision boundary

3.1.1. Objective function

First, we leverage the structure of the consistent objective function of uPU to develop a consistent objective function for UC-PUAL. Let us compare the objective function of PUAL in Eq. (5) and that of uPU in Eq. (1). We note that: firstly, $\frac{1}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+$ in Eq. (5) can be regarded as corresponding to $\hat{L}_p^1(f)$ in Eq. (1); secondly, $\frac{1}{n_u}[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]^T[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]$ in Eq. (5) can be regarded as $\hat{L}_u^{-1}(f)$ in Eq. (1). Hence, we need to introduce into Eq. (5) a new term - $\frac{1}{n_p}[\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]^T[\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]$, which can be regarded as corresponding to $-\hat{L}_p^{-1}(f)$ in Eq. (1), i.e., to borrow the objective function's idea and structure of uPU into the objective function of PUAL. The obtained objective function of UC-PUAL can be initially formulated as

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{\pi c}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ \\ + \frac{c}{n_u}[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)]^T[\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)] \\ - \frac{\pi c}{n_p}[\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]^T[\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)] \\ + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0),\end{aligned} \tag{6}$$

where c is a hyper-parameter and π is the class prior as in uPU.

Second, we revise the third and fourth terms with the absolute loss in Eq. (6) to ensure universal consistency with the Bayes classifier. This adjustment allows the loss function to be expressed in the form of Eq. (A.6), which is required for universal consistency. Further details are provided in Appendix A.1. In this way, the objective function of UC-PUAL with linear decision boundary can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{\pi c}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ + \frac{c}{n_u}\|\mathbf{1}_u + (\boldsymbol{X}_{[u]}\boldsymbol{\beta} + \mathbf{1}_u\beta_0)\|_1 \\ - \frac{\pi c}{n_p}\|\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)\|_1 + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0).\end{aligned} \tag{7}$$

3.1.2. Parameter estimation

The asymmetric loss of the objective function in Eq. (7) cannot always meet the linear-odd condition proposed in [31, Lemma 4], i.e.,

$$l(f,1) - l(f,-1) = [1-f]_+ - |1+f| = \begin{cases} 2 \neq -2f, f < -1, \\ -2f, -1 < f < 1, \\ -f \neq -2f, f \geq 1. \end{cases} \tag{8}$$

Not satisfying the odd condition can render the objective function in Eq. (7) non-convex, leading to significant challenges in optimisation [23].

Despite ADMM being initially proposed for convex optimisation in [32], in recent years studies [33] have explored the convergence conditions of ADMM for non-convex and non-differentiable objective functions. In this section, we propose an algorithm based on ADMM for the non-convex optimisation of UC-PUAL in Eq. (7).

Firstly, let us slightly rewrite the objective function in Eq. (7), to meet the convergence conditions in [33, Table 1]. Let matrix

$$\boldsymbol{C}_n = \begin{bmatrix} -\frac{\pi c}{n_p}\boldsymbol{I}_p & \mathbf{0} \\ \mathbf{0} & \frac{c}{n_u}\boldsymbol{I}_u, \end{bmatrix} \tag{9}$$

where $\boldsymbol{I}_u$ is an $n_u \times n_u$ identity matrix and $\boldsymbol{I}_p$ is an $n_p \times n_p$ identity matrix. In this case, the objective function of UC-PUAL in Eq. (7) can be transformed to the following three-block form:

$$\begin{aligned}&\min_{\boldsymbol{\beta},\beta_0,\boldsymbol{h},\boldsymbol{a}} \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0) + \frac{\pi c}{n_p}\mathbf{1}_p^T[\boldsymbol{h}]_+ + \mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++} \\ &s.t.\ \boldsymbol{h} = \mathbf{1}_p - (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0),\ \boldsymbol{a} = \mathbf{1}_{pu} + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0),\end{aligned} \tag{10}$$

where $[\boldsymbol{a}]_{++}$ is a column vector and the ith element of $[\boldsymbol{a}]_{++}$ is $|a_i|$

The objective function in Eq. (10) can be divided into three blocks, i.e., $\frac{\pi c}{n_p}\mathbf{1}_p^T[\boldsymbol{h}]_+$, $\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}$ and $\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)$:

1. $\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + (\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)$ is convex and Lipschitz differentiable w.r.t. $\boldsymbol{\beta}$ and β_0.
2. $\frac{\pi c}{n_p}\mathbf{1}_p^T[\boldsymbol{h}]_+$ is convex but not always differentiable w.r.t. $\boldsymbol{h}$.
3. $\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}$ is neither convex nor always differentiable w.r.t. $\boldsymbol{a}$.

We also would like to make two remarks:

- $\mathbf{1}_p^T[\boldsymbol{h}]_+ = \sum_{i=1}^{n_p}\max(0,h_i)$ and $\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++} = \frac{c}{n_u}\sum_{i=n_p+1}^{n_{pu}}|a_i| - \frac{\pi c}{n_p}\sum_{i=1}^{n_p}|a_i|$; this indicates that $\mathbf{1}_p^T[\boldsymbol{h}]_+$ and $\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}$ are piece-wise linear functions for $\boldsymbol{h}$ and $\boldsymbol{a}$, respectively.
- Furthermore, $\frac{\pi c\partial\mathbf{1}_p^T[\boldsymbol{h}]_+}{n_p\partial\boldsymbol{h}}$ is a column vector consisting of elements that are either $\frac{\pi c}{n_p}$ or 0; $\frac{\partial\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}}{\partial\boldsymbol{a}}$ is a column vector whose elements take value from $\{\frac{\pi c}{n_p},-\frac{\pi c}{n_p},\frac{c}{n_u},-\frac{c}{n_u}\}$; this indicates that $\frac{\pi c\partial\mathbf{1}_p^T[\boldsymbol{h}]_+}{n_p\partial\boldsymbol{h}}$ and $\frac{\partial\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}}{\partial\boldsymbol{a}}$ are bounded in any bounded set.

Then, we note that, according to [33, Table 1], for the non-convex objective function to be solved via ADMM, the non-convex blocks and the blocks not always differentiable are required to be piece-wise linear and their partial derivatives are required to be bounded in any bounded set. Moreover, [33] also requires the convex blocks to be Lipschitz differentiable. Hence, as discussed above, the three blocks $\mathbf{1}_p^T[\boldsymbol{h}]_+$, $\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}$ and $\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta}+(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)$ all meet their corresponding requirements. Therefore, based on the structure of ADMM in [33], we propose the following algorithm to solve the optimisation of the three-block form of UC-PUAL in Eq. (10).

Firstly, the Lagrangian function of the objective function of UC-PUAL in Eq. (10) can be written as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_{uc}) =& \frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta}+(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)\\ &+\frac{\pi c}{n_p}\mathbf{1}_p^T[\boldsymbol{h}]_+ + \mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++} + \boldsymbol{u_h}^T[\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}+\mathbf{1}_p\beta_0)-\boldsymbol{h}]\\ &+\boldsymbol{u_a}^T(\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0-\boldsymbol{a})\\ s.t.\ &\boldsymbol{h}=\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}+\mathbf{1}_p\beta_0),\\ &\boldsymbol{a}=\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0,\end{aligned} \tag{11}$$

where $\boldsymbol{\theta}_{uc}=\{\boldsymbol{\beta},\beta_0,\boldsymbol{h},\boldsymbol{a},\boldsymbol{u_h},\boldsymbol{u_a}\}$, $\boldsymbol{u_h}$ and $\boldsymbol{u_a}$ are dual variables.

Then, the augmented Lagrangian function of UC-PUAL is defined as

$$\begin{aligned}\mathcal{L}_a(\boldsymbol{\theta}_{uc}) =& \mathcal{L}(\boldsymbol{\theta}_{uc})+\frac{\mu_1}{2}\left\|\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}+\mathbf{1}_p\beta_0)-\boldsymbol{h}\right\|_2^2\\ &+\frac{\mu_2}{2}\left\|\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0-\boldsymbol{a}\right\|_2^2.\end{aligned} \tag{12}$$

Update of $\boldsymbol{a}$. Then, following Eq. (12), the update of $\boldsymbol{a}$ is to solve

$$\begin{aligned}\boldsymbol{a}^{(k+1)} =& \arg\min_{\boldsymbol{a}}\frac{1}{\mu_2}\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}+\frac{\boldsymbol{u_a}^{(k)^T}}{\mu_2}(\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}^{(k)}+\mathbf{1}_{pu}\beta_0^{(k)}-\boldsymbol{a})\\ &+\frac{1}{2}\left\|\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}^{(k)}+\mathbf{1}_{pu}\beta_0^{(k)}-\boldsymbol{a}\right\|_2^2.\end{aligned} \tag{13}$$

This is equivalent to optimise

$$\arg\min_{\boldsymbol{a}}\frac{1}{\mu_2}\mathbf{1}_{pu}^T\boldsymbol{C}_n[\boldsymbol{a}]_{++}+\frac{1}{2}\left\|\mathbf{1}_{pu}+\frac{\boldsymbol{u_a}^{(k)}}{\mu_2}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}^{(k)}+\mathbf{1}_{pu}\beta_0^{(k)}-\boldsymbol{a}\right\|_2^2. \tag{14}$$

As the terms containing $a_i, i=1,\dots,n_{pu}$ in Eq. (14) do not contain other elements of $\boldsymbol{a}$, we can solve the update of $a_1^{(k+1)},\dots,a_{n_{pu}}^{(k+1)}$ independently.

That is, firstly for $a_i^{(k+1)}, i=1,\dots,n_p$, the objective function is

$$\frac{-\pi c}{\mu_2 n_p}|a_i|+\frac{1}{2}\left(1+\frac{u_{ai}^{(k)}}{\mu_2}+\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)}-a_i\right)^2. \tag{15}$$

To minimise Eq. (15), we can consider the following function w.r.t. x:

$$j_p|x|+\frac{1}{2}(x-d_p)^2, j_p<0, \tag{16}$$

where j_p and d_p are constants. There are two cases of the threshold function in Eq. (16), thus we can define

$$g_{j_p}^{[1]}(d_p)=\arg\min_x j_p|x|+\frac{1}{2}(x-d_p)^2=\begin{cases}d_p+j_p, & d_p<0,\\ d_p-j_p, & d_p\geq 0.\end{cases} \tag{17}$$

Therefore the solution of $a_i^{(k+1)}, i=1,\dots,n_p$ can be obtained via computing

$$a_i^{(k+1)}=g_{\frac{-\pi c}{\mu_2 n_p}}^{[1]}(1+\frac{u_{ai}^{(k)}}{\mu_2}+\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)}), i=1,2.,\dots,n_p. \tag{18}$$

Then for the update of $a_i^{(k+1)}, i=n_p+1,\dots,n_{pu}$, we need to separately solve

$$\frac{c}{\mu_2 n_u}|a_i|+\frac{1}{2}\left(1+\frac{u_{ai}^{(k)}}{\mu_2}+\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)}-a_i\right)^2. \tag{19}$$

To minimise Eq. (19) we can consider the following function w.r.t. x:

$$j_u|x|+\frac{1}{2}(x-d_u)^2, j_u>0, \tag{20}$$

where j_u and d_u are constants. The three cases of the threshold function in Eq. (20) are as follows:

$$\arg\min_x j_u|x|+\frac{1}{2}(x-d_u)^2=\begin{cases}d_u+j_u, & d_u<-j_u,\\ 0, & -j_u\leq d_u\leq j_u,\\ d_u-j_u, & d_u<-j_u.\end{cases} \tag{21}$$

Thus, by defining $g_{j_u}^{[2]}(d_u)=\arg\min_x j_u|x|+\frac{1}{2}(x-d_u)^2$, $a_i^{(k+1)}, i=n_p+1,\dots,n_{pu}$ can be solved via computing

$$a_i^{(k+1)}=g_{\frac{c}{\mu_2 n_p}}^{[2]}(1+\frac{u_{ai}^{(k)}}{\mu_2}+\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)}), i=n_p+1,\dots,n_p+n_u. \tag{22}$$

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

$$\begin{aligned}\boldsymbol{h}^{(k+1)} =& \arg\min_{\boldsymbol{h}}\frac{\pi c}{n_p}\mathbf{1}_p^T[\boldsymbol{h}]_+ + \boldsymbol{u_h}^{(k)^T}[\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}^{(k)}+\mathbf{1}_p\beta_0^{(k)})-\boldsymbol{h}]\\ &+\frac{\mu_1}{2}\left\|\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}^{(k)}+\mathbf{1}_p\beta_0^{(k)})-\boldsymbol{h}\right\|_2^2,\end{aligned} \tag{23}$$

which is equivalent to solve the problem

$$\min_{\boldsymbol{h}}\sum_{i=1}^{n_p}\left\{\frac{\pi c}{n_p\mu_1}[h_i]_+ + \frac{1}{2}[1+\frac{u_{hi}^{(k)}}{\mu_1}-(\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)})-h_i]^2\right\}. \tag{24}$$

To minimise the threshold function in Eq. (24), suppose function $j[x]_+ + \frac{1}{2}(x-d)^2, j>0$ and

$$s_j(d)=\arg\min_x j[x]_+ + \frac{1}{2}(x-d)^2=\begin{cases}d-j, d>j,\\ 0, 0\leq d\leq j,\\ d, d<0.\end{cases} \tag{25}$$

Then h_i, $i=1,\dots,n_p$, can be updated via computing

$$h_i^{(k+1)}=s_{\frac{\pi c}{n_p}}\left[1+\frac{u_{hi}^{(k)}}{\mu_1}-(\boldsymbol{x}_i^T\boldsymbol{\beta}^{(k)}+\beta_0^{(k)})\right]. \tag{26}$$

Update of $\boldsymbol{\beta}$ *and* β_0. The update of $\boldsymbol{\beta}$ and β_0 is to solve

$$\begin{aligned}&\arg\min_{\boldsymbol{\beta},\beta_0}\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta}+(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0)\\ &+\boldsymbol{u_h}^{(k)^T}[\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}+\mathbf{1}_p\beta_0)-\boldsymbol{h}^{(k+1)}]\\ &+\frac{\mu_1}{2}\left\|\mathbf{1}_p-(\boldsymbol{X}_{[p]}\boldsymbol{\beta}+\mathbf{1}_p\beta_0)-\boldsymbol{h}^{(k+1)}\right\|_2^2\\ &+\boldsymbol{u_a}^{(k)^T}[\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0-\boldsymbol{a}^{(k+1)}]\\ &+\frac{\mu_2}{2}\left\|\mathbf{1}_{pu}+\boldsymbol{X}_{[pu]}\boldsymbol{\beta}+\mathbf{1}_{pu}\beta_0-\boldsymbol{a}^{(k+1)}\right\|_2^2,\end{aligned} \tag{27}$$

which is a quadratic function. Therefore we can solve the optimisation in Eq. (27) via the Karush–Kuhn–Tucker (KKT) conditions directly.

Let $\boldsymbol{I}_k, \forall k \in \mathbb{Z}$ denote a $k \times k$ identity matrix. By defining

$$\begin{aligned}
&\boldsymbol{M}_{11} = \lambda \boldsymbol{I}_m + 2\boldsymbol{X}_{[pu]}^T \boldsymbol{R} \boldsymbol{X}_{[pu]} + \mu_1 \boldsymbol{X}_{[p]}^T \boldsymbol{X}_{[p]} + \mu_2 \boldsymbol{X}_{[pu]}^T \boldsymbol{X}_{[pu]},\\
&\boldsymbol{M}_{12} = 2\boldsymbol{X}_{[pu]}^T \boldsymbol{R} \boldsymbol{1}_{pu} + \mu_1 \boldsymbol{X}_{[p]}^T \boldsymbol{1}_p + \mu_2 \boldsymbol{X}_{[pu]}^T \boldsymbol{1}_{pu},\\
&\boldsymbol{M}_{21} = \boldsymbol{M}_{12}^T,\\
&M_{22} = 2\boldsymbol{1}_{pu}^T \boldsymbol{R} \boldsymbol{1}_{pu} + \mu_1 n_p + \mu_2 (n_p + n_u),\\
&\boldsymbol{m}_1 = \boldsymbol{X}_{[p]}^T \boldsymbol{u}_{\boldsymbol{h}}^{(k)} + \mu_1 \boldsymbol{X}_{[p]}^T (\boldsymbol{1}_p - \boldsymbol{h}^{(k+1)}) - \boldsymbol{X}_{[pu]}^T \boldsymbol{u}_{\boldsymbol{a}}{}^{(k)} - \mu_2 \boldsymbol{X}_{[pu]}^T (\boldsymbol{1}_{pu} - \boldsymbol{a}^{(k+1)}),\\
&m_2 = \boldsymbol{1}_p^T \boldsymbol{u}_{\boldsymbol{h}}^{(k)} + \mu_1 \boldsymbol{1}_p^T (\boldsymbol{1}_p - \boldsymbol{h}^{(k+1)}) - \boldsymbol{1}_{pu}^T \boldsymbol{u}_{\boldsymbol{a}}{}^{(k)} - \mu_2 \boldsymbol{1}_{pu}^T (\boldsymbol{1}_{pu} - \boldsymbol{a}^{(k+1)}),
\end{aligned} \tag{28}$$

the solution of problem in Eq. (27) can be obtained by solving the following linear equation w.r.t. $\boldsymbol{\beta}$ and β_0:

$$\begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}^{(k+1)} \\ \beta_0^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{m}_1 \\ m_2 \end{bmatrix}. \tag{29}$$

Update of $\boldsymbol{u_h}$ and $\boldsymbol{u_a}$. Following [33], the update of $\boldsymbol{u_h}$ and $\boldsymbol{u_a}$ can be obtained as

$$\begin{aligned}
&\boldsymbol{u_h}^{(k+1)} = \boldsymbol{u_h}^{(k)} + \mu_1 [\boldsymbol{1}_p - (\boldsymbol{X}_{[p]} \boldsymbol{\beta}^{(k+1)} + \boldsymbol{1}_p \beta_0^{(k+1)}) - \boldsymbol{h}^{(k+1)}],\\
&\boldsymbol{u_a}^{(k+1)} = \boldsymbol{u_a}^{(k)} + \mu_2 [\boldsymbol{1}_{pu} + \boldsymbol{X}_{[pu]} \boldsymbol{\beta}^{(k+1)} + \boldsymbol{1}_{pu} \beta_0^{(k+1)} - \boldsymbol{a}^{(k+1)}].
\end{aligned} \tag{30}$$

3.1.3. Algorithm

The algorithm of UC-PUAL with linear decision boundary can be summarised in Algorithm 1.

Algorithm 1 UC-PUAL with linear decision boundary

Input: PU dataset, c, λ, σ and μ_1
Output: $\boldsymbol{\beta}$ and β_0

Initialise $\boldsymbol{\beta}$, β_0, $\boldsymbol{h}$, $\boldsymbol{a}$, $\boldsymbol{u}_h$, $\boldsymbol{u}_a$
while not converged **do**
  Update $\boldsymbol{a}^{(k+1)} = \arg\min_{\boldsymbol{a}} \mathcal{L}_a(\boldsymbol{\beta}^{(k)}, \beta_0^{(k)}, \boldsymbol{h}^{(k)}, \boldsymbol{a}, \boldsymbol{u_h}^{(k)}, \boldsymbol{u_a}^{(k)})$
  Update $\boldsymbol{h}^{(k+1)} = \arg\min_{\boldsymbol{h}} \mathcal{L}_a(\boldsymbol{\beta}^{(k)}, \beta_0^{(k)}, \boldsymbol{h}, \boldsymbol{a}^{(k+1)}, \boldsymbol{u_h}^{(k)}, \boldsymbol{u_a}^{(k)})$
  Update $(\boldsymbol{\beta}^{(k+1)}, \beta_0^{(k+1)}) = \arg\min_{\boldsymbol{\beta}, \beta_0} \mathcal{L}_a(\boldsymbol{\beta}, \beta_0, \boldsymbol{h}^{k+1}, \boldsymbol{a}^{(k+1)}, \boldsymbol{u_h}^{(k+1)}, \boldsymbol{u_a}^{(k)})$
  Update $\boldsymbol{u_h}^{(k+1)} = \boldsymbol{u_h}^{(k)} + \mu_1 [\boldsymbol{1}_p - (\boldsymbol{X}_{[p]} \boldsymbol{\beta}^{(k+1)} + \boldsymbol{1}_p \beta_0^{(k+1)}) - \boldsymbol{h}^{(k+1)}]$
  Update $\boldsymbol{u_a}^{(k+1)} = \boldsymbol{u_a}^{(k)} + \mu_2 [\boldsymbol{1}_{pu} + \boldsymbol{X}_{[pu]} \boldsymbol{\beta}^{(k+1)} + \boldsymbol{1}_{pu} \beta_0^{(k+1)} - \boldsymbol{a}^{(k+1)}]$
end while

3.2. UC-PUAL with non-linear decision boundary

In this section, we develop a kernel-based algorithm to solve UC-PUAL with non-linear decision boundary. The way of using kernel trick here is similar to those used by previous methods [15,34–36].

3.2.1. Objective function

Suppose $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^{M \times 1}$ be a mapping of the instance vector $\boldsymbol{x}$. Then let $\boldsymbol{\phi}(\boldsymbol{X}_{[k]}) \in \mathbb{R}^{n_k \times r}, k = p, u, pu$ be the mapping of the original data matrix $\boldsymbol{X}_{[k]}$. The ith row of $\boldsymbol{\phi}(\boldsymbol{X}_{[k]})$ is $\phi(\boldsymbol{x}_i)^T \in \mathbb{R}^{1 \times r}$. According to Eqs. (28) and (29), once we substitute $\boldsymbol{X}_{[pu]}$ for $\boldsymbol{\phi}(\boldsymbol{X}_{pu})$ during the training of classifiers, the following necessary condition for the optimal solution of $\boldsymbol{\beta}$ to satisfy can be obtained: $\boldsymbol{\beta} = \boldsymbol{B}^{-1} \boldsymbol{\phi}(\boldsymbol{X}_{[pu]})^T \boldsymbol{\Omega}$, where $\boldsymbol{B} = \boldsymbol{M}_{11} - \dfrac{\boldsymbol{M}_{12} \boldsymbol{M}_{21}}{M_{22}}$, and

$$\begin{aligned}
\boldsymbol{\Omega} = &\begin{bmatrix} \boldsymbol{u_h} - \mu_1 (\boldsymbol{1}_p - \boldsymbol{h}) - \mu_1 \frac{m_2}{M_{22}} \boldsymbol{1}_p \\ 0 \end{bmatrix} - [\boldsymbol{u_a} + \mu_2 (\boldsymbol{1}_{pu} - \boldsymbol{a}) + \frac{2m_2}{M_{22}} \boldsymbol{R} \boldsymbol{1}_{pu} \\
&+ \frac{\mu_2}{M_{22}} \boldsymbol{1}_{pu}].
\end{aligned} \tag{31}$$

Therefore, the predictive score function in Eq. (4) for instance $\boldsymbol{x}^*$ of UC-PUAL can be transformed to

$$f = \boldsymbol{\Phi}(\boldsymbol{x}^*, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \beta_0. \tag{32}$$

Define kernel matrices $\boldsymbol{\Phi}(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]}) = \boldsymbol{\phi}(\boldsymbol{X}_{[k]}) \boldsymbol{B}^{-1} \boldsymbol{\phi}(\boldsymbol{X}_{[pu]})^T$ and $\boldsymbol{\Phi}_2(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]}) = \boldsymbol{\phi}(\boldsymbol{X}_{[k]}) \boldsymbol{B}^{-1} \boldsymbol{B}^{-1} \boldsymbol{\phi}(\boldsymbol{X}_{[pu]})^T$ for $k = p, u$, or pu. Then, the objective function of UC-PUAL with kernel trick applied can be represented as

$$\begin{aligned}
\min_{\boldsymbol{\Omega}, \beta_0} &\frac{\lambda}{2} \boldsymbol{\Omega}^T \boldsymbol{\Phi}_2(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \frac{\pi c}{n_p} \boldsymbol{1}_p^T [\boldsymbol{1}_p - (\boldsymbol{\Phi}(\boldsymbol{X}_{[p]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \boldsymbol{1}_p \beta_0)]_+ \\
&+ \frac{c}{n_u} \|\boldsymbol{1}_u + \boldsymbol{\Phi}(\boldsymbol{X}_{[u]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \beta_0 \boldsymbol{1}_u\|_1 - \frac{\pi c}{n_p} \|\boldsymbol{1}_p + \boldsymbol{\Phi}(\boldsymbol{X}_{[p]}, \boldsymbol{X}_{[p]}) \boldsymbol{\Omega} + \beta_0 \boldsymbol{1}_p\|_1 \\
&+ (\boldsymbol{\Phi}(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \boldsymbol{1}_{pu} \beta_0)^T \boldsymbol{R} (\boldsymbol{\Phi}(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega} + \boldsymbol{1}_{pu} \beta_0),
\end{aligned} \tag{33}$$

whose solution is not related to $\boldsymbol{X}_{[k]}$, $k = p, u$, or pu once the kernel matrices are determined.

3.2.2. Parameter estimation

In this case, the update of $\boldsymbol{a}$ can be written as

$$a_i^{(k+1)} = \begin{cases} g^{[1]}_{\frac{\pi c}{\mu_2 n_p}} \left[1 + \frac{u_{ai}^{(k)}}{\mu_2} + \boldsymbol{\Phi}(\boldsymbol{x}_i, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k)} + \beta_0^{(k)} \right], & i = 1, \ldots, n_p, \\ g^{[2]}_{\frac{c}{\mu_2 n_p}} \left[1 + \frac{u_{ai}^{(k)}}{\mu_2} + \boldsymbol{\Phi}(\boldsymbol{x}_i, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k)} + \beta_0^{(k)} \right], & i = n_p + 1, \ldots, n_p + n_u. \end{cases} \tag{34}$$

Then, the update of $\boldsymbol{h}$ can be reformulated as

$$h_i^{(k+1)} = s_{\frac{\pi c}{n_p}} \left[1 + \frac{u_{hi}^{(k)}}{\mu_1} - (\boldsymbol{\Phi}(\boldsymbol{x}_i, \boldsymbol{X}_{[pu]}) \Omega^{(k)} + \beta_0^{(k)}) \right], i = 1, \ldots, n_p. \tag{35}$$

Then, we can update β_0 via

$$\beta_0^{(k+1)} = \frac{m_2}{M_{22}} - \boldsymbol{Q}_b^{(k+1)} / M_{22}, \tag{36}$$

where m_2, M_{22} are not related to $\boldsymbol{X}_{[p]}, \boldsymbol{X}_{[u]}, \boldsymbol{X}_{[pu]}$ and

$$\begin{aligned}
\boldsymbol{Q}_b^{(k+1)} = &(2\boldsymbol{1}_{pu}^T \boldsymbol{R} + \mu_2 \boldsymbol{1}_{pu}^T) \boldsymbol{\Phi}(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k+1)} \\
&+ \mu_1 \boldsymbol{1}_p^T \boldsymbol{\Phi}(\boldsymbol{X}_{[p]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k+1)}.
\end{aligned} \tag{37}$$

Finally, the update of $\boldsymbol{u_h}$ and $\boldsymbol{u_a}$ becomes

$$\begin{aligned}
&\boldsymbol{u_h}^{(k+1)} = \boldsymbol{u_h}^{(k)} + \mu_1 [\boldsymbol{1}_p - (\boldsymbol{\Phi}(\boldsymbol{X}_{[p]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k+1)} + \boldsymbol{1}_p \beta_0^{(k+1)}) - \boldsymbol{h}^{(k+1)}],\\
&\boldsymbol{u_a}^{(k+1)} = \boldsymbol{u_a}^{(k)} + \mu_2 [\boldsymbol{1}_{pu} + \boldsymbol{\Phi}(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]}) \boldsymbol{\Omega}^{(k+1)} + \boldsymbol{1}_{[pu]} \beta_0^{(k+1)} - \boldsymbol{a}^{(k+1)}].
\end{aligned} \tag{38}$$

We note that $\boldsymbol{\Phi}_2(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]})$ does not directly appear in the update process for the optimisation in this section so that we only need to determine the form of $\boldsymbol{\Phi}(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]})$. Moreover, λ also does not appear directly in the update process; it is contained in matrix $\boldsymbol{B}$ as a part of $\boldsymbol{\Phi}(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]})$. Therefore, for convenience, we use λ to represent the hyper-parameter(s) of the kernel matrix $\boldsymbol{\Phi}(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]})$.

3.2.3. Algorithm

The algorithm of UC-PUAL with non-linear decision boundary can be summarised in Algorithm 2.

Algorithm 2 UC-PUAL with non-linear decision boundary

Input: PU dataset, $\boldsymbol{\Phi}$, c, λ, σ and μ_1
Output: $\boldsymbol{\Omega}$ and β_0

Initialise $\boldsymbol{\Omega}$, β_0, $\boldsymbol{h}$, $\boldsymbol{a}$, $\boldsymbol{u}_h$ and $\boldsymbol{u}_a$.
while not converged **do**
  Update $\boldsymbol{a}$ via Equation (34)
  Update $\boldsymbol{h}$ via Equation (35)
  Update $\boldsymbol{\Omega}$ and via Equation (31) w.r.t. $\boldsymbol{h}^{(k+1)}$, $\boldsymbol{a}^{(k+1)}$, $\boldsymbol{u_h}^{(k)}$ and $\boldsymbol{u_a}^{(k)}$
  Update β_0 via Equation (36)
  Update $\boldsymbol{u_h}$ and $\boldsymbol{u_a}$ via Equation (38)
end while

3.3. Universal consistency

3.3.1. Bayes risk

Firstly we define the risk, i.e., the expected error rate, of a binary classifier with decision function $f^*(\boldsymbol{x}) = \text{sgn}(f(x)) \in \{-1, 1\}$ as

$$\mathcal{R}(f^*) = \int_{(\boldsymbol{x},y)\in S} \beth(f^*(\boldsymbol{x}) \neq y) P(\boldsymbol{X} = \boldsymbol{x}, Y = y) d\boldsymbol{x} dy = P[f^*(\boldsymbol{X}) \neq Y], \tag{39}$$

where $\beth(\cdot)$ is the indicator function; S is the domain of the instance $(\boldsymbol{X}, Y)$; and $\mathcal{R}$ indicates the probability of a classifier to misclassify instance $(\boldsymbol{X}, Y)$ selected at random from this domain.

Let S_x be the domain of $\boldsymbol{X}$. We can divide S_x into the following three regions by the class which instance $(\boldsymbol{X} = \boldsymbol{x}, Y = y)$ is more likely to belong to:

$$\begin{aligned} \boldsymbol{Z}_+ &= \{\boldsymbol{x} \in S_x : P(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}) > P(Y = -1 \mid \boldsymbol{X} = \boldsymbol{x})\}, \\ \boldsymbol{Z}_- &= \{\boldsymbol{x} \in S_x : P(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}) < P(Y = -1 \mid \boldsymbol{X} = \boldsymbol{x})\}, \\ \boldsymbol{Z}_0 &= \{\boldsymbol{x} \in S_x : P(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}) = P(Y = -1 \mid \boldsymbol{X} = \boldsymbol{x})\}. \end{aligned} \tag{40}$$

Then, based on the three regions $\boldsymbol{Z}_+$, $\boldsymbol{Z}_-$ and $\boldsymbol{Z}_0$ in Eq. (40), the Bayes decision function f^*_{Bayes} can be defined as 1 for $\boldsymbol{x} \in \boldsymbol{Z}_+ \cup \boldsymbol{Z}_0$ and -1 for $\boldsymbol{x} \in \boldsymbol{Z}_-$. Hence, the misclassification probability $\eta(\boldsymbol{x}) = P(Y = -1|\boldsymbol{X} = \boldsymbol{x})$ for $\boldsymbol{x} \in \boldsymbol{Z}_+ \cup \boldsymbol{Z}_0$ and $\eta(\boldsymbol{x}) = P(Y = 1|\boldsymbol{X} = \boldsymbol{x})$ for $\boldsymbol{x} \in \boldsymbol{Z}_-$.

The classifier with the Bayes decision function is called the Bayes classifier, and the risk of the Bayes classifier is termed the Bayes risk, which can be obtained in our case from Eq. (39) as

$$\mathcal{R}_{\text{Bayes}} = \mathcal{R}(f^*_{\text{Bayes}}) = \int_{\boldsymbol{x}\in S_x} \eta(\boldsymbol{x}) P(\boldsymbol{X} = \boldsymbol{x}) d\boldsymbol{x}. \tag{41}$$

3.3.2. Universal consistency of UC-PUAL

Suppose that the feature mapping $\boldsymbol{\phi}(\cdot)$ is used to train UC-PUAL and define the covering number $\mathcal{N}\left(\left(S_x, d_{\boldsymbol{\phi}}\right), \epsilon\right)$, where metric $d_{\boldsymbol{\phi}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{\phi}(\boldsymbol{x}_i) - \boldsymbol{\phi}(\boldsymbol{x}_j)\|_2^2$, to be the minimum number of hyper-spheres with diameter $\epsilon > 0$ to cover the entire metric space $(S_x, d_{\boldsymbol{\phi}})$. Then according to [37, Lemma 1], the universal kernel $\boldsymbol{\Phi}^*(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{\phi}(\boldsymbol{x}_1)^T \boldsymbol{\phi}(\boldsymbol{x}_2)$ specifies the kernel functions when $\boldsymbol{\phi}(\cdot)$ is continuous and $\forall \epsilon > 0$, $\mathcal{N}\left(\left(S_x, d_{\boldsymbol{\phi}}\right), \epsilon\right)$ can be regarded as a finite function w.r.t. ϵ.

Define $c' = \frac{2c}{\lambda}$ and the decision function of UC-PUAL trained from sample size n_{pu} to be $f_{\text{uc}}^{*n_{pu}}$. Then the universal consistency of UC-PUAL can be summarised by the following theorem:

Theorem 1. *Firstly, suppose that S_x is compact, and $\boldsymbol{\Phi}(\boldsymbol{X}_{[k]}, \boldsymbol{X}_{[pu]})$ is a universal kernel function. Secondly, suppose that there exists constant $\alpha > 0$ satisfying $\mathcal{N}\left(\left(S_x, d_{\boldsymbol{\phi}}\right), \epsilon\right) \in \mathcal{O}(\epsilon^{-\alpha})$. Thirdly, suppose that there exists constant δ satisfying $0 < \delta < \frac{1}{\alpha}$, and when n_p and n_u tend to infinity, the value of c' also tends to infinity with $c' \in \mathcal{O}(n_{pu}^{\delta})$. In this case, $\forall \epsilon > 0$, we have*

$$P^{n_{pu}}\left[\mathcal{R}\left(f_{uc}^{*n_{pu}}\right) - \mathcal{R}_{Bayes} \leqslant \epsilon\right] \to 1,$$

where $\mathcal{R}\left(f_{uc}^{*n_{pu}}\right)$ is the risk of the trained decision function $f_{uc}^{*n_{pu}}$ of UC-PUAL at sample size n_{pu}.

Theorem 1 indicates that, with the size of the training set increasing, the gap between the Bayes risk and the risk of UC-PUAL tends to 0 in probability. The proof of Theorem 1 is in Appendix. Notably, the effectiveness of the universal consistency can be approximated via the inequality in Theorem 2 in Appendix A.2.

4. Experiments

4.1. Experiments on synthetic datasets

In this section, experiments were conducted on linearly separable synthetic datasets to verify the superiority of UC-PUAL compared with PUAL and GLLC. The generation of synthetic positive–negative datasets is the same as that in [19].

Table 1
Summary of the average F1-score (%) and the standard deviation of the experiments on the synthetic datasets; The best result of each row is in blue. When UC-PUAL achieves the highest accuracy, * indicates the p-value from the Wilcoxon signed-rank test comparing UC-PUAL with the second-best method, with significance levels denoted as *** for $p < 0.01$, ** for $p < 0.05$, and * for $p < 0.1$.

$\textbf{mean}_{p2}$	UC-PUAL	PUAL	GLLC
50	96.41 ± 1.43*	93.26 ± 1.80	91.08 ± 2.74
100	96.67 ± 1.48*	93.15 ± 0.98	85.52 ± 5.87
200	96.03 ± 1.63**	94.25 ± 1.60	81.09 ± 9.01
500	97.02 ± 1.52**	92.55 ± 2.07	74.64 ± 6.78
1000	96.97 ± 1.87**	91.51 ± 0.84	71.67 ± 6.16

4.1.1. Training–test split for the synthetic PU datasets

Considering UC-PUAL was proposed in the case-control scenario [9], we split each of the generated synthetic datasets to construct the PU training and test sets consistent with the case-control scenario by the following two steps:

1. γ' of the synthetic positive instances were picked randomly into the labelled-positive set, and the rest positive instances were into the unlabelled set.
2. The whole labelled-positive set and 70% of the unlabelled set formed the training set. The rest 30% of the unlabelled set formed the test set.

In this case, we obtained 25 pairs of PU training set and test set. Moreover, γ' is set to $\frac{7}{37}$, and we have the training sets with label frequency $\gamma = \gamma'/(0.3\gamma' + 0.7) = 0.25$.

4.1.2. Model setting

For the hyper-parameter tuning of UC-PUAL, PUF-score was used:

$$\text{PUF} - \text{score} = \frac{\text{recall}^2}{P[\text{sgn}(f(\boldsymbol{x})) = 1]}, \tag{42}$$

where 'recall' is to be estimated by $\frac{1}{n_p}\sum_{\boldsymbol{x}_i\in p} \beth(\text{sgn}(f(\boldsymbol{x}_i)) = 1)$ with the indicator function denoted by $\beth(\cdot)$, and $P[\text{sgn}(f(\boldsymbol{x})) = 1]$ can be estimated via $\frac{1}{n_u}\sum_{\boldsymbol{x}_i\in u} \beth(\text{sgn}(f(\boldsymbol{x}_i)) = 1)$. Firstly, c was set to n_u and the number K of the k-nearest neighbours was set to 5 for generating local similarity matrix $\boldsymbol{R}$. Then σ and λ were picked from the set $\{1, 2, 3, 4, 5\} \circ \{0.1, 1, 10, 100\}$ and tuned by 4-fold cross-validation (CV). Parameters λ and σ in GLLC and PUAL were tuned in the same way as in UC-PUAL. For GLLC and PUAL, c_u was tuned from the set $\{0.01, 0.02, \ldots, 0.5\} \circ n_u$ while c_p was set to n_p.

4.1.3. Results and analysis

The results of the experiments, on the constructed synthetic PU datasets, are summarised in Table 1. The results are measured by the average F1-score.

From Table 1, we can observe the following patterns. Firstly, UC-PUAL always has better performance than PUAL and GLLC on the synthetic PU datasets with all the 5 values of $\textbf{mean}_{p2}$. This indicates that UC-PUAL can have better performance to generate the linear decision boundary than PUAL and GLLC on trifurcate data when class prior π is known. Secondly, with the $\textbf{mean}_{p2}$ become larger, the improvement offered by UC-PUAL over PUAL and GLLC becomes larger. To assess the statistical significance of the improvement, we conducted a right-tailed Wilcoxon signed-rank test on the F1-score difference between UC-PUAL and the second-best method, with $\text{H}_0 : M_D \leq 0$, and $\text{H}_1 : M_D > 0$, where M_D is the median of difference. In all scenarios, UC-PUAL demonstrates a statistically significant improvement over the second-best method PUAL.

Table 2
The average F1-score (%) with the standard deviation of the classifiers trained on the 16 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. The rest of the caption is as in Table 1.

Dataset	UC-PUAL	PUAL	GLLC	uPU	nnPU	Robust-PU	T-HOneCls
OR1	93.0 ± 3.2***	90.1 ± 2.3	85.6 ± 3.8	16.6 ± 33.3	84.1 ± 6.9	85.8 ± 2.3	82.4 ± 6.4
	87.4 ± 2.5***	83.9 ± 5.8	72.9 ± 5.5	20.9 ± 33.1	72.1 ± 7.0	80.6 ± 4.9	78.4 ± 3.7
OR2	88.3 ± 1.6	88.9 ± 1.2	86.5 ± 1.4	76.9 ± 4.9	81.6 ± 4.2	84.2 ± 4.7	83.7 ± 5.8
	86.1 ± 2.5**	85.5 ± 3.4	77.1 ± 5.7	74.4 ± 5.5	77.3 ± 3.8	81.0 ± 2.2	80.5 ± 2.5
Pen	98.9 ± 1.2***	92.5 ± 8.1	88.9 ± 10.2	77.8 ± 31.0	87.5 ± 14.9	93.7 ± 3.8	91.6 ± 2.9
	98.2 ± 1.9***	91.7 ± 9.0	87.0 ± 11.4	72.6 ± 31.0	84.1 ± 16.9	91.8 ± 2.6	88.2 ±3.5
Seeds	94.6 ± 1.9	92.3 ± 4.9	94.6 ± 2.8	92.4 ± 1.5	97.3 ± 3.7	100 ± 0.0	100 ± 0.0
	94.3 ± 1.9	89.1 ± 5.5	91.2 ± 4.5	86.9 ± 3.1	93.1 ± 3.9	100 ± 0.0	100 ± 0.0
HD	88.1 ± 2.5**	82.7 ± 2.4	82.0 ± 5.5	71.4 ± 4.2	74.4 ± 2.2	85.3 ± 5.0	83.7 ± 7.9
	87.8 ± 2.6***	81.9 ± 4.0	84.5 ± 4.1	71.0 ± 4.0	75.1 ± 2.4	80.6 ± 1.3	81.4 ± 5.2
Acc	62.1 ± 3.3	65.0 ± 4.8	68.1 ± 2.2	20.1 ± 27.6	20.5 ± 28.6	72.5 ±4.9	76.1 ±7.4
	60.0 ± 7.6	66.4 ± 4.4	64.1 ± 3.3	22.0 ± 29.6	23.4 ± 31.4	69.1 ±5.3	71.9 ±8.1
OD	100 ± 0.0***	89.0 ± 8.4	100 ± 0.0	80.0 ± 42.2	100 ± 0.0	100 ±0.0	100 ±0.0
	100 ± 0.0***	95.7 ± 6.7	100 ± 0.0	80.0 ± 42.2	100 ± 0.0	100 ±0.0	100 ±0.0
PB	98.7 ± 1.9	95.9 ± 1.1	100 ± 0.0	69.8 ± 2.6	67.2 ± 3.2	96.2± 1.7	98.2 ±2.7
	99.1 ± 0.6	97.9 ± 0.7	100 ± 0.0	68.8 ± 2.6	66.6 ± 4.1	93.1 ± 2.5	96.9 ±3.2
Ecoli	90.3 ± 1.3	90.8 ± 2.6	88.6 ± 2.8	84.4 ± 6.1	85.9 ± 6.7	86.2 ± 2.4	88.3 ± 9.8
	89.3 ± 1.3	88.0 ± 4.4	89.4 ± 3.7	84.9 ± 6.8	86.1 ± 6.6	83.1± 1.8	85.5 ± 11.0
SSMCR	87.9 ± 0.9	87.6 ± 1.3	87.8 ± 1.8	85.7 ± 2.0	87.4 ± 1.4	62.8 ± 0.9	60.7 ± 8.4
	87.9 ± 0.9	87.6 ± 1.3	87.5 ± 1.6	85.0 ± 2.0	86.8 ± 1.5	62.5 ± 0.9	69.8 ± 6.1
ENB	55.8 ± 9.3	42.8 ± 4.8	42.7 ± 4.6	29.6 ± 22.1	30.2 ± 23.7	69.3 ±5.2	64.8 ± 3.2
	53.9 ± 9.5	45.8 ± 7.5	44.2 ± 6.6	26.1 ± 30.5	26.9 ± 31.3	65.1 ± 3.1	64.0 ±7.1
LD	53.4 ± 4.5	44.2 ± 5.7	50.8 ± 6.9	11.9 ± 25.8	31.5 ± 27.8	83.9 ± 3.7	86.0 ± 1.5
	56.4 ± 4.4	36.9 ± 10.0	40.1 ± 8.9	10.2 ± 22.4	20.1 ± 26.3	81.1 ±6.1	80.9 ± 1.9
UMD	98.4 ± 1.6	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0
	99.0 ± 1.5	99.6 ± 0.9	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0
RD	83.1 ± 2.3*	82.5 ± 2.2	83.1 ± 2.9	70.6 ± 12.9	71.3 ± 13.6	62.8 ± 0.9	60.7 ± 8.4
	80.9 ± 5.3	77.6 ± 3.8	81.2 ± 2.3	72.9 ± 14.5	73.1 ± 12.8	62.6 ± 2.4	69.8 ± 8.1
MNIST1	89.7 ± 4.1***	86.5 ± 3.9	80.4 ± 3.8	81.9 ± 2.5	83.5 ± 4.4	87.1 ± 3.0	85.9 ± 1.6
	88.3 ± 3.7***	84.2 ± 5.2	79.5 ± 1.9	77.2 ± 4.1	82.1 ± 3.9	85.5 ± 3.6	84.7 ± 2.8
MNIST2	84.6 ± 4.3	81.7 ± 4.0	85.2 ± 3.9	84.0 ± 2.6	85.5 ± 4.5	89.3 ± 3.1	87.9 ± 1.7
	80.6 ± 3.8	79.3 ± 5.3	81.7 ± 2.0	79.3 ± 5.2	84.2 ± 4.6	87.2 ± 2.9	85.3 ± 4.1

4.2. Experiments on real-world data

4.2.1. Real-world datasets

Fourteen real-world datasets from the UCI Machine Learning Repository were used to assess the performance of UC-PUAL: Pen-Based Recognition of Handwritten Digits (**Pen**), Accelerometer (**Acc**), User Knowledge Modelling Data Set (**UMD**), **Seeds**, Liver Disorders (**LD**), **Ecoli**, Parking Birmingham (**PB**), Sepsis survival minimal clinical records (**SSMCR**), Raisin Dataset (**RD**), Occupancy Detection (**OD**), Online Retail (**OR1**), Online Retail II (**OR2**), Energy efficiency Data Set (**ENB**) and Heart Disease (**HD**). We also constructed two image datasets, trifurcate **MNIST1** and non-trifurcate **MNIST2**, from MNIST[1] [38]. **MNIST1** treats digits 1 and 8 as positive and 7 as negative, while **MNIST2** treats 1 and 7 as positive and 8 as negative. [Details on how the datasets are constructed as PU data can be found in [19].

4.2.2. Compared methods and model setting

As the compared methods with UC-PUAL, PUAL, GLLC, uPU, nnPU, Robust-PU and T-HOneCls were also trained on the 16 real-world datasets. GLLC and PUAL serve as the baseline of UC-PUAL; uPU and nnPU are two consistent PU learning methods; Robust-PU [5] and T-HOneCls [39] are two recent state-of-the-arts for multi-step approach and one-step approach, respectively.

The setting of parameters here is similar to that in Section 4.1.2, except that λ and σ were first tuned from the set $\{10^{-4}, 10^{-3}, \ldots, 10^{3}, 10^{4}\}$ based on the setting in [15], and then continually tuned following a greedy algorithm based on the average PUF-score on the validation sets as follows:

1. Set λ, σ from the grid search.
2. Update one of λ, σ by increasing/decreasing 10% of its current value. The optimal case in the 4-fold CV is treated as the update of this step.
3. Repeat Step 2 until there is no better case of λ and σ.

For GLLC and PUAL, λ, σ were tuned in the same way. Similarly, c_u was firstly tuned from the set $\{0.01, 0.02, \ldots, 0.5\} \circ n_u$ and then tuned with the above greedy algorithm. The hyper-parameters of uPU and nnPU were as recommended by [17].[2] The hyper-parameters of Robust-PU were as recommended by [5].[3] The hyper-parameters of T-HOneCls were as recommended by [39][4]. Radial Basis Function (RBF) kernel $\exp\left(-\|x_i - x_j\|^2/2\lambda^2\right)$ for the (i,j)th element of kernel matrix $\boldsymbol{\Phi}(\boldsymbol{X}_{[pu]}, \boldsymbol{X}_{[pu]})$ was applied to UC-PUAL, PUAL and GLLC.

4.2.3. Results and analysis

The Training–Test split of the datasets is the same as that in Section 4.1.1. The results of the experiments are summarised in Table 2 by average F1-score for the 32 cases of 16 real-world datasets. We can make the following observations. Firstly, UC-PUAL achieved very competitive or superior F1-score than PUAL on 26/32 cases, including 7/8

[1] https://github.com/cvdfoundation/mnist?tab=readme-ov-file.

[2] https://github.com/kiryor/nnPUlearning.

[3] https://github.com/woriazzc/robust-pu.

[4] https://github.com/Hengwei-Zhao96/T-HOneCls.

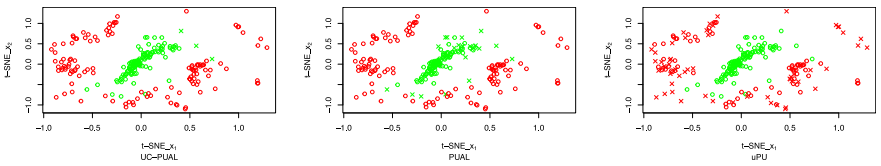


Fig. 2. The t-SNE plots of the test classification results of UC-PUAL, PUAL and uPU on the trifurcate dataset, **Pen**; red: positive instances; green: negative instances; cross: incorrectly classified; circle: correctly classified. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

cases of the four trifurcate datasets **Pen**, **OR1**, **OR2** and **MNIST1**; this indicates that the universally consistent objective function is of benefit to the PU classification on trifurcate datasets. Secondly, uPU and nnPU often exhibited much larger standard deviations than PUAL and UC-PUAL. This is potentially because their algorithms, based on Adam for optimising their non-convex objective functions, cannot always converge to the optimal solution. Thirdly, in 12/32 cases, UC-PUAL achieved the best performance among all the seven methods compared in the experiments. Finally, whenever UC-PUAL achieves the highest accuracy, it is consistently statistically significantly better than the second-best method, as confirmed by the Wilcoxon signed-rank test.

For **Acc** and **UMD**, we observe a slight reduction in classification accuracy of UC-PUAL compared with PUAL. This is likely because the sample sizes of the two datasets are limited. For **Acc**, **PB** and **RD**, **MNIST2**, UC-PUAL and PUAL are worse than GLLC, indicating that the distributions of these datasets are not trifurcate.

The t-SNE visualisations of the correctly and incorrectly classified instances on one test set of the trifurcate dataset, **Pen**, for UC-PUAL, PUAL and uPU, are shown in Fig. 2. Clearly, UC-PUAL achieves the best classification, with minimal false negatives (green crosses), while PUAL has more false negatives concentrated on the bottom-left corner, and uPU exhibits many false positives (red crosses).

5. Conclusion and future work

In this paper, we propose UC-PUAL, a universally consistent PU classifier to achieve better classification on trifurcate PU datasets, where positive instances distribute on both sides of negative instances. The key novelty of UC-PUAL is to integrate the idea of PUAL, which uses an asymmetric structure of loss on positive instances, and the idea of uPU, which offers a consistent setting of PU classification. The superiority of UC-PUAL was demonstrated by experiments on both synthetic and real-world datasets.

The performance of UC-PUAL is heavily dependent on the estimation of the class prior π. In future work, we aim to propose a consistent objective function without estimating π. Furthermore, UC-PUAL is specifically designed to make PUAL universally consistent. In the future, we aim to design a more general universally consistent framework for different PU classifiers.

CRediT authorship contribution statement

Xiaoke Wang: Writing – original draft, 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.

Appendix. Proof of universal consistency of UC-PUAL

A.1. The approximate of UC-PUAL

Let us consider the objective function of UC-PUAL in Eq. (A.1) with kernel mapping $\boldsymbol{\phi}(X_{[k]}), k = p, u, pu$ substituting $\boldsymbol{X}_{[p]}$, $\boldsymbol{X}_{[u]}$, and $\boldsymbol{X}_{[pu]}$ respectively.

$$\begin{aligned}\min_{\boldsymbol{\beta},\beta_0} &\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{\pi c}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{\phi}(\boldsymbol{X}_{[p]})\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ \\ &+ \frac{c}{n_u}\|\mathbf{1}_u + (\boldsymbol{\phi}(\boldsymbol{X}_{[u]})\boldsymbol{\beta} + \mathbf{1}_u\beta_0)\|_1 \\ &- \frac{\pi c}{n_p}\|\mathbf{1}_p + (\boldsymbol{\phi}(\boldsymbol{X}_{[p]})\boldsymbol{\beta} + \mathbf{1}_p\beta_0)\|_1 + (\boldsymbol{\phi}(\boldsymbol{X}_{[pu]})\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{\phi}(\boldsymbol{X}_{[pu]})\boldsymbol{\beta} \\ &+ \mathbf{1}_{pu}\beta_0).\end{aligned} \tag{A.1}$$

According to [15], the local constraint $(\boldsymbol{\phi}(\boldsymbol{X}_{[pu]})\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)^T\boldsymbol{R}(\boldsymbol{\phi}(\boldsymbol{X}_{[pu]})\boldsymbol{\beta} + \mathbf{1}_{pu}\beta_0)$ in the objective function of UC-PUAL with kernel mapping in Eq. (A.1) can be transformed to

$$\frac{2}{n_{pu}}\sum_{\boldsymbol{x}_i,\boldsymbol{x}_j \text{ are KNN of each other}} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma}\right)(f(\boldsymbol{\phi}(\boldsymbol{x}_i)) - f(\boldsymbol{\phi}(\boldsymbol{x}_j)))^2, \tag{A.2}$$

where $f(\boldsymbol{\phi}(\boldsymbol{x})) = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\beta} + \beta_0$.

For certain $i = 1, \ldots, n_{pu}$, define the r.v. $\mathcal{X}_{k[i]}^{[mx]}$ to be the kth maximum value of set $\{(-\|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2^2) : j = 1, \ldots, i-1, i+1, \ldots, n_{pu}.\}$ of $n_{pu} - 1$ elements. Hence,

$$P[\mathcal{X}_{1[i]}^{[mx]} \le x] = \mathcal{P}_{i-}^{n_{pu}-1}(x), \tag{A.3}$$

where $\mathcal{P}_{i-}(x)$ is the cdf of any r.v. $-\|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2^2$ for certain i.

Here we make an assumption that $\forall x < 0$, there is $\mathcal{P}_{i-}(x) < 1$; hence, as n_{pu} tends to infinity, $P[\mathcal{X}_{1[i]}^{[mx]} \le x]$ tends to 0 for $x < 0$. Meanwhile, only the pdf $p[\mathcal{X}_{1[i]}^{[mx]} = 0]$ tends to infinity as n_{pu} tends to infinity, so that $\mathcal{X}_{1[i]}^{[mx]}$ converges to 0 in probability.

Generally for $k \ge 1$, we have

$$\begin{aligned}P[\mathcal{X}_{k[i]}^{[mx]} \le x] &= \sum_{a=n_{pu}-k}^{n_{pu}-1}\binom{n_{pu}-1}{a}\mathcal{P}_{i-}^{a}(x)(1-\mathcal{P}_{i-}(x))^{n_{pu}-1-a} \\ &= \mathcal{P}_{i-}^{n_{pu}-1}(x) + \sum_{a=2}^{k}\binom{n_{pu}-1}{a-1}(1-\mathcal{P}_{i-}(x))^{a-1}\mathcal{P}_{i-}^{n_{pu}-a}(x).\end{aligned} \tag{A.4}$$

Note that, for limited k and thus a, $0 < \binom{n_{pu}-1}{a-1}\mathcal{P}_{i-}^{n_{pu}-a}(x) \le n_{pu}^{a-1}\mathcal{P}_{i-}^{n_{pu}-a}(x)$. As $n_{pu}^{a-1}\mathcal{P}_{i-}^{n_{pu}-a}(x)$ converges to 0 as n_{pu} tends to infinity, we can justify that $P[\mathcal{X}_{k[i]}^{[mx]} \le x]$ converges to 0 for $x < 0$, as n_{pu} tends to infinity; that is, $\mathcal{X}_{k[i]}^{[mx]}$ converges to 0 in probability. Furthermore, the case appearing with $\mathcal{X}_{k[i]}^{[mx]} = 0$ is continuous function $(f(\boldsymbol{\phi}(\boldsymbol{x}_i)) - f(\boldsymbol{\phi}(\boldsymbol{x}_j)))^2 = 0$ for $\boldsymbol{x}_j$ to be the kth nearest neighbour of $\boldsymbol{x}_i$. Hence the local constraint in Eq. (A.2) can be regarded as the weighted average of the r.v.s converging to 0 in probability. Therefore, the local constraint also converges to 0 in probability as n_{pu} tends to infinity.

It follows that we only need to consider $\frac{\lambda}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{\pi c}{n_p}\mathbf{1}_p^T[\mathbf{1}_p - (\boldsymbol{\phi}(\boldsymbol{X}_{[p]})\boldsymbol{\beta} + \mathbf{1}_p\beta_0)]_+ + \frac{c}{n_u}\|\mathbf{1}_u + (\boldsymbol{\phi}(\boldsymbol{X}_{[u]})\boldsymbol{\beta} + \mathbf{1}_u\beta_0)\|_1 - \frac{\pi c}{n_p}\|\mathbf{1}_p + (\boldsymbol{\phi}(\boldsymbol{X}_{[p]})\boldsymbol{\beta} + \mathbf{1}_p\beta_0)\|_1$ in the objective function of UC-PUAL for sufficiently large n_{pu}. This objective function is consistent with the following one by introducing the term $-\frac{\pi c}{n_p}\|\mathbf{1}_p + (\boldsymbol{X}_{[p]}\boldsymbol{\beta} + \mathbf{1}_p\beta_0)\|_1$:

$$\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{c'}{n_u}\sum_{\boldsymbol{x}_i\in\boldsymbol{X}_{[u]}} l(f(\boldsymbol{\phi}(\boldsymbol{x}_i);\boldsymbol{\beta},\beta_0),y_i) \tag{A.5}$$

where $c' = \frac{2c}{\lambda}$. Because the use of the absolute loss in the objective function in PUAL in Eq. (7), the asymmetric loss function in Eq. (A.5) can be represented as

$$l(f(\boldsymbol{x};\boldsymbol{\beta},\beta_0),y) = \begin{cases} [1-\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\beta}-\beta_0]_+, & y=1; \\ |1+\boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{\beta}+\beta_0|, & y=-1. \end{cases} \tag{A.6}$$

This loss function is important for our latter proof of contradictory, with details provided in Appendix A.4.2.

The predictive score function of this approximate of UC-PUAL is the same as the one of GLLC in Eq. (4). In this case, $\mathcal{R}\left(f_{\text{uc}}^{*n_u}\right) \to \mathcal{R}\left(f_{\text{ap}}^{*n_u}\right)$ with n_u increasing, where $\mathcal{R}\left(f_{\text{ap}}^{*n_u}\right)$ is the risk of the trained decision function $f_{\text{ap}}^{*n_u}$ of the approximate of UC-PUAL in Eq. (A.5) with $c' = c_{n_u}$ and the sample size n_u.

Then we can find the kernel form of the optimisation of Eq. (A.5) via the KKT conditions w.r.t. $\boldsymbol{\beta}$ as

$$\begin{aligned} &\min_{\boldsymbol{v},\beta_0,\boldsymbol{\xi}} \frac{1}{2}\boldsymbol{v}^T\boldsymbol{\Phi}^*(\boldsymbol{X}_{[u]},\boldsymbol{X}_{[u]})\boldsymbol{v} + \frac{c'}{n_u}\sum_{i=1}^{n_u}\xi_i \\ &s.t.\ \xi_i \geq 1 - \boldsymbol{\Phi}^*(\boldsymbol{X}_{[u]},\boldsymbol{X}_{[u]})\boldsymbol{v} - \beta_0, \text{for } i \text{ with } y_i = 1; \xi_i \geq 0, \text{for } i \text{ with } y_i = 1; \\ &\quad \xi_i \geq 1 + \boldsymbol{\Phi}^*(\boldsymbol{X}_{[u]},\boldsymbol{X}_{[u]})\boldsymbol{v} + \beta_0, \text{for } i \text{ with } y_i = -1; \\ &\quad \xi_i \geq -1 - \boldsymbol{\Phi}^*(\boldsymbol{X}_{[u]},\boldsymbol{X}_{[u]})\boldsymbol{v} - \beta_0, \text{for } i \text{ with } y_i = -1. \end{aligned} \tag{A.7}$$

where the (i,j) element of kernel matrix $\boldsymbol{\Phi}^*(\boldsymbol{X}_{[u]},\boldsymbol{X}_{[u]})$ is $\boldsymbol{\Phi}^*(\boldsymbol{x}_i,\boldsymbol{x}_j)$.

Furthermore, for certain $\boldsymbol{B}$ with continuous $\boldsymbol{\phi}(\cdot)\boldsymbol{B}^{-1/2}$ and finite $\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{B}^{-1/2}\boldsymbol{\phi}}\right),\epsilon\right)$, we can also find $\boldsymbol{\phi}(\cdot) = \boldsymbol{\phi}(\cdot)\boldsymbol{B}^{-1/2}\boldsymbol{B}^{1/2}$ continuous with $\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{\phi}}\right),\epsilon\right)$ finite. Therefore, $\boldsymbol{\Phi}^*(\boldsymbol{x}_1,\boldsymbol{x}_2) = \boldsymbol{\phi}(\boldsymbol{x}_1)^T\boldsymbol{\phi}(\boldsymbol{x}_2)$ can be regarded as a universal kernel once $\boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{B}^{-1}\boldsymbol{\phi}(\boldsymbol{x})^T$ is a universal kernel.

A.2. Universal consistency of the approximate

To prove Theorem 1, firstly we prove the universal consistency of the approximate of UC-PUAL with the objective function in Eq. (A.7), following the idea in [40, pp. 775–776], which proves the universal consistency of the classic SVM. We can give the following Theorem 2 for the approximate of UC-PUAL:

Theorem 2. *Suppose $\mathcal{S}_x$ is compact and the kernel function $\Phi(\cdot)$ is universal. $\forall 0 < \epsilon < 1$, we can find a constant $c^* > 0$ such that for all $c' \geq c^*$ there is*

$$\begin{aligned} &P^{n_u}\left[\mathcal{R}\left(f_{\text{ap}}^{*n_u}\right) - \mathcal{R}_{Bayes} \leq \epsilon\right] \geq 1 - 2Me^{-\frac{\epsilon^6 n_u}{2^{29}M^2}}, \\ &M = \frac{64}{\epsilon}\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{\Phi}}\right), \frac{\epsilon}{32\sqrt{c'}}\right). \end{aligned}$$

A.3. Step 2: Construction of a 'Representative' dataset

In this section, we construct a 'representative' dataset based on the domain $\mathcal{S}_x$ itself, which is independent of the objective function, the loss function and the predictive score function. Therefore, what we do in this section is the same as the corresponding part in [40, pp. 776–780]. In this case, we summarise the important details of [40] with the proof (referring to the proof of Lemma 2 to Lemma 4 in [40]) skipped and then provide some additional analysis.

Firstly we can divide $\mathcal{S}_x$ into the following subsets:

$$\mathcal{S}_{x[i]} = \begin{cases} \{\boldsymbol{x}\in\mathcal{S}_x : i2^{-\rho} \leq \eta(\boldsymbol{x}) < (i+1)2^{-\rho}\}, & i = 0,1,\ldots,2^{\rho-1}-2, \\ \{\boldsymbol{x}\in\mathcal{S}_x : i2^{-\rho} \leq \eta(\boldsymbol{x}) \leq \frac{1}{2}\} & i = 2^{\rho-1}-1. \end{cases} \tag{A.8}$$

where ρ is the integer meeting $2^{-\rho} \leq \tau \leq 2^{-\rho+1}$ and $\tau = \epsilon/32$; this leads to the following relationship:

$$\begin{aligned} &\sum_{i=0}^{2^{\rho-1}-1}\frac{i}{2^\rho}P[\boldsymbol{X}\in\mathcal{S}_{x[i]}] \leq \mathcal{R}_{\text{Bayes}} \leq \sum_{i=0}^{2^{\rho-1}-1}\frac{i}{2^\rho}P[\boldsymbol{X}\in\mathcal{S}_{x[i]}] + \frac{1}{2^\rho}\sum_{i=0}^{2^{\rho-1}-1}P[\boldsymbol{X}\in\mathcal{S}_{x[i]}] \\ &\quad \leq \sum_{i=0}^{2^{\rho-1}-1}\frac{i}{2^\rho}P[\boldsymbol{X}\in\mathcal{S}_{x[i]}] + \tau. \end{aligned} \tag{A.9}$$

To control the numbers of the positive and negative instances in the 'representative' dataset, we need to divide $\mathcal{S}_{x[i]}, i = 0,1,\ldots,2^{\rho-1}-2$, into $\mathcal{S}_{x[i]}^1 = \mathcal{S}_{x[i]} \cap \boldsymbol{Z}_+$ and $\mathcal{S}_{x[i]}^{-1} = \mathcal{S}_{x[i]} \cap \boldsymbol{Z}_-$. Furthermore, we can construct a 'large' enough compact subset $\mathcal{B}_{[i]}^j$ of $\mathcal{S}_{x[i]}^j$, i.e.,

$$P\left[\boldsymbol{X}\in\mathcal{S}_{x[i]}^j \backslash \mathcal{B}_{[i]}^j\right] \leq \tau 2^{-\rho}, \quad i = 0,\ldots,2^{\rho-1}-2, j\in\{-1,1\}. \tag{A.10}$$

Furthermore, there exists subset $\mathcal{B}_{[2^{\rho-1}-1]}$ of $\mathcal{S}_{x[2^{\rho-1}-1]}$ meeting

$$P\left[\boldsymbol{X}\in\mathcal{S}_{x[2^{\rho-1}-1]}\backslash\mathcal{B}_{[2^{\rho-1}-1]}\right] \leq \tau 2^{-\rho} \tag{A.11}$$

For convenience, let $\mathcal{B}_{[2^{\rho-1}-1]}^1 = \mathcal{B}_{[2^{\rho-1}-1]} \cap (\boldsymbol{Z}_+ \cup \boldsymbol{Z}_0)$ and $\mathcal{B}_{[2^{\rho-1}-1]}^{-1} = \mathcal{B}_{[2^{\rho-1}-1]} \cap \boldsymbol{Z}_-$.

As proved in Lemma 2 of [40], when $\boldsymbol{\Phi}^*(\boldsymbol{X}_1,\boldsymbol{X}_2) = \boldsymbol{\phi}(\boldsymbol{X}_1)\boldsymbol{\phi}(\boldsymbol{X}_2)^T$ to be universal kernel, there exists value $\tilde{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ to satisfy

$$\begin{aligned} &\boldsymbol{\phi}(\boldsymbol{x})^T\tilde{\boldsymbol{\beta}} \in [1,1+\tau],\ \boldsymbol{x}\in\cup_{i=0}^{2^{\rho-1}-2}\mathcal{B}_{[i]}^1; \boldsymbol{\phi}(\boldsymbol{x})^T\tilde{\boldsymbol{\beta}} \in [-(1+\tau),-1], \\ &\quad \boldsymbol{x}\in\cup_{i=0}^{2^{\rho-1}-2}\mathcal{B}_{[i]}^{-1}; \\ &\boldsymbol{\phi}(\boldsymbol{x})^T\tilde{\boldsymbol{\beta}} \in [-\tau,\tau],\ \boldsymbol{x}\in\mathcal{B}_{[2^{\rho-1}-1]}; \boldsymbol{\phi}(\boldsymbol{x})^T\tilde{\boldsymbol{\beta}} \in [-(1+\tau),1+\tau], \\ &\quad \boldsymbol{x}\notin\cup_{j=-1,1}\cup_{i=0}^{2^{\rho-1}-1}\mathcal{B}_{[i]}^j. \end{aligned} \tag{A.12}$$

Formulae in (A.12) are used to construct the upper bound of the contradiction in Appendix A.4.1.

Let $\sigma = \tau/\sqrt{c'}$. For $i = 0,\ldots,2^{\rho-1}-1$ and $j = -1,1$, we are able to divide $\mathcal{B}_{[i]}^j$ into finite partition $\tilde{\mathbb{A}}_i^j$ with the diameter of each set $\mathcal{A}\in\tilde{\mathbb{A}}_i^j$ no greater than σ in the kernel space. According to the definition of covering number, the cardinality of $\tilde{\mathbb{A}}_i^j$ is no greater than $\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{\phi}}\right),\sigma\right)$. Based on this, we can define

$$\mathbb{A}_i^j = \left\{\mathcal{A}\in\tilde{\mathbb{A}}_i^j : P[\boldsymbol{X}\in\mathcal{A}] \geq \frac{2\tau}{M}\right\}, \tag{A.13}$$

with $2^\rho \leq |\cup_{j=-1,1}\cup_{i=0}^{2^{\rho-1}-1}\mathbb{A}_i^j| \leq M$. Therefore, recalling $M = \frac{64}{\epsilon}\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{\Phi}}\right), \frac{\epsilon}{32\sqrt{c'}}\right)$, there is

$$\begin{aligned} &\sum_{\mathcal{A}\in\mathbb{A}_i^j} P[\boldsymbol{X}\in\mathcal{A}] = P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j] - P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j \backslash \cup_{\mathcal{A}\in\mathbb{A}_i^j}\mathcal{A}] \\ &\geq P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j] - \frac{2\tau}{M}\mathcal{N}\left(\left(\mathcal{S}_x, d_{\boldsymbol{\phi}}\right),\sigma\right) = P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j] - \frac{2\tau}{M}\frac{\tau}{2}M \\ &= P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j] - \tau^2 \geq P[\boldsymbol{X}\in\mathcal{B}_{[i]}^j] - \tau. \end{aligned} \tag{A.14}$$

For convenience, let $\mathcal{B}_{[i]}^{*j} = \cup_{\mathcal{A}\in\mathbb{A}_i^j}\mathcal{A}$ for $i = 0,\ldots,2^{\rho-1}-1, j\in\{-1,1\}$.

Consider the following conditions for the dataset $\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_{n_u},y_{n_u})\}$ with $n_u \gg 2^{\rho+1}$:

$$\begin{aligned} F_{n_u,\mathcal{A}}^+ &= \left\{\left((\boldsymbol{x}_1,y_1),\ldots,(\boldsymbol{x}_{n_u},y_{n_u})\right) : \left|\{l : \boldsymbol{x}_l\in\mathcal{A}, y_l = j\}\right| \right. \\ &\quad \left. \geq n_u(1-\tau)\left(1-\frac{i+1}{2^\rho}\right)P[\boldsymbol{X}\in\mathcal{A}]\right\}, \\ F_{n_u,\mathcal{A}}^- &= \left\{\left((\boldsymbol{x}_1,y_1),\ldots,(\boldsymbol{x}_{n_u},y_{n_u})\right) : \left|\{i : \boldsymbol{x}_l\in\mathcal{A}, y_i \neq j\}\right| \right. \\ &\quad \left. \geq n_u(1-\tau)\frac{i}{2^\rho}P[\boldsymbol{X}\in\mathcal{A}]\right\}, \end{aligned} \tag{A.15}$$

where $i = 0, \ldots, 2^{\rho-1} - 2, j \in \{-1, 1\}$ and $\mathcal{A} \in \mathbb{A}_i^j$. Besides, for $\mathcal{A} \in \mathbb{A}_{2^{\rho-1}-1}^j, j \in \{-1, 1\}$ we can define the conditions as

$$\begin{aligned} F_{n_u,\mathcal{A}}^{+} = \Bigg\{ \Big((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\Big) : \big|\{l : \boldsymbol{x}_l \in \mathcal{A}, y = 1\}\big| \\ \geq n_u(1-\tau)\left(\frac{1}{2} - \frac{1}{2^\rho}\right) P[\boldsymbol{X} \in \mathcal{A}] \Bigg\}. \\ F_{n_u,\mathcal{A}}^{-} = \Bigg\{ \Big((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\Big) : \big|\{l : \boldsymbol{x}_l \in \mathcal{A}, y = -1\}\big| \\ \geq n_u(1-\tau)\left(\frac{1}{2} - \frac{1}{2^\rho}\right) P[\boldsymbol{X} \in \mathcal{A}] \Bigg\}. \end{aligned} \tag{A.16}$$

We need to ensure a minimum number of instances form each set $\mathcal{A} \in \mathbb{A}_i^j$, $i = 0, \ldots, 2^{\rho-1} - 1, j \in \{-1, 1\}$ to construct the 'representative' dataset. More specifically, let $F_{n_u} = \cap_{j\in\{-1,1\}} \cap_{i=0}^{2^{\rho-1}-1} \cap_{\mathcal{A}\in\mathbb{A}_i^j} \left(F_{n_u,\mathcal{A}}^{+} \cap F_{n_u,\mathcal{A}}^{-}\right)$. We can construct the 'representative' dataset as we make the dataset meet both Condition (A.15) and Condition (A.16), i.e., $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\} \in F_{n_u}$. The probability of obtaining such a 'representative' dataset via i.i.d. sampling form the population is

$$P^{n_u}\left(F_{n_u}\right) \geq 1 - 2Me^{-2(\tau^6/M^2)n_u} = 1 - 2Me^{-\frac{\epsilon^6 n_u}{2^{29}M^2}}, \tag{A.17}$$

for $n_u \gg 2^{\rho+1}$ as proved in Lemma 3 of [40]. There are at least 2^ρ positive instances and negative instances in the 'representative' dataset since $P[\boldsymbol{X} \in \mathcal{A}]$ in Eq. (A.15) and Eq. (A.16) is always greater than 0 according to Eq. (A.13),

A.4. Step 3: Proof of Theorem 2 by contradiction

In this section, we prove that once $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\}$ are the 'representative' instances, we will have $\mathcal{R}\left(f_{\text{ap}}^{*n_u}\right) - \mathcal{R}_{\text{Bayes}} < \epsilon$, via the proof by contradiction in an inequality.

A.4.1. Upper bound of the inequality for contradiction

Firstly, assume that there is a 'representative' dataset $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\} \in F_n$ with

$$\mathcal{R}\left(f_{\text{ap}}^{*n_u}\right) - \mathcal{R}_{\text{Bayes}} > \epsilon. \tag{A.18}$$

Define $\boldsymbol{\beta}_{[\text{ap}]}$ and $\beta_{[\text{ap}]0}$ to be the optimal solution of the objective function in Eq. (A.7). Then let the value of the slack variable of instance $(\boldsymbol{x}_l, y_l)$ in Eq. (A.7) of instance $(\boldsymbol{x}_l, y_l)$, with $\boldsymbol{\beta} = \boldsymbol{\beta}_{[\text{ap}]}$ and $\beta_0 = \beta_{[\text{ap}]0}$, to be $\xi_{[\text{ap}]l}$. Similarly, let the value of the slack variable of instance $(\boldsymbol{x}_l, y_l)$, with $\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}$ and $\beta_0 = 0$, to be $\tilde{\xi}_l$.

Furthermore, according to the relationships in Eq. (A.12) and the constraints in Eq. (A.7), there are the following eight scenarios for $\tilde{\xi}_l$: for $x_l \in \mathcal{B}_{[i]}^{*1}$ and $y_l = 1$, let $\tilde{\xi}_l = 0$; for $x_l \in \mathcal{B}_{[i]}^{*1}$ and $y_l = -1$, let $\tilde{\xi}_l = 2+\tau$; For $x_l \in \mathcal{B}_{[i]}^{*-1}$ and $y_l = 1$, let $\tilde{\xi}_l = 2 + \tau$; for $x_l \in \mathcal{B}_{[i]}^{*-1}$ and $y_l = -1$, let $\tilde{\xi}_l = \tau$; for $x_l \in \mathcal{B}_{[2^{\rho-1}-1]}$ and $y_l = 1$, let $\tilde{\xi}_l = 1 + \tau$; for $x_l \in \mathcal{B}_{[2^{\rho-1}-1]}$ and $y_l = -1$, let $\tilde{\xi}_l = 1 + \tau$; for $x_l \notin (\cup_{i=0}^{2^{\rho-1}-2} \cup_{j=\{-1,1\}} \mathcal{B}_{[i]}^{*j}) \cup \mathcal{B}_{[2^{\rho-1}-1]}$ and $y_l = 1$, let $\tilde{\xi}_l = 2 + \tau$; for $x_l \notin (\cup_{i=0}^{2^{\rho-1}-2} \cup_{j=\{-1,1\}} \mathcal{B}_{[i]}^{*j}) \cup \mathcal{B}_{[2^{\rho-1}-1]}$ and $y_l = -1$, let $\tilde{\xi}_l = 2 + \tau$.

Then let $n_1, n_1^+, n_1^-, n_2, n_3, n_4$ denote the number of specific instances in the 'representative' set $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_{n_u}, y_{n_u})\}$ as

$$\begin{aligned} n_1^+ &= \left|\left\{l : \boldsymbol{x}_l \in \cup_{i=0}^{2^{\rho-1}} \mathcal{B}_{[i]}^{*1}, y_l = 1\right\}\right|, n_1^- = \left|\left\{l : \boldsymbol{x}_l \in \cup_{i=0}^{2^{\rho-1}} \mathcal{B}_{[i]}^{*-1}, y_l = -1\right\}\right|, \\ n_2 &= \left|\left\{l : \boldsymbol{x}_l \in \cup_{i=0}^{2^{\rho-1}} \mathcal{B}_{[i]}^{*-1}, y_l = 1\right\}\right| + \left|\left\{l : \boldsymbol{x}_l \in \cup_{i=0}^{2^{\rho-1}-2} \mathcal{B}_{[i]}^{*1}, y_l = -1\right\}\right|, \\ n_1 &= n_1^+ + n_1^-, n_3 = \left|\left\{l : \boldsymbol{x}_l \in \mathcal{B}_{[2^{\rho-1}-1]}\right\}\right|, \\ n_4 &= \left|\left\{l : \boldsymbol{x}_l \notin (\cup_{i=0}^{2^{\rho-1}-2} \cup_{j=\{-1,1\}} \mathcal{B}_{[i]}^{*j}) \cup \mathcal{B}_{[2^{\rho-1}-1]}\right\}\right|. \end{aligned} \tag{A.19}$$

According to Eq. (A.19), $n_u = n_1 + n_2 + n_3 + n_4$. Furthermore, as $(\boldsymbol{\beta}_{[\text{ap}]}, \beta_{[\text{ap}]0})$ is the optimal solution of $(\boldsymbol{\beta}, \beta_0)$, we have

$$\begin{aligned} \boldsymbol{\beta}_{[\text{ap}]}^T \boldsymbol{\beta}_{[\text{ap}]} + \frac{c'}{n_u}\sum_{l=1}^{n_u} \xi_{[\text{ap}]l} &\leq \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{\beta}} + \frac{c'}{n_u}\sum_{l=1}^{n_u} \tilde{\xi}_l \leq \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{\beta}} + \frac{c'}{n_u}\left(\tau n_1^- + (2+\tau)n_2\right. \\ &\quad \left. +(1+\tau)n_3 + (2+\tau)n_4\right) \\ &= \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{\beta}} + \frac{c'}{n_u}\left(\tau n_1^- + (2+\tau)\left(n_u - n_1\right) - n_3\right). \end{aligned} \tag{A.20}$$

Then according to Inequality (A.14) and the inequality condition in (A.15), the same as the content in [40] (Proof of Lemma 4), there is

$$\begin{aligned} (2+\tau)\left(n_u - n_1\right) \leq \Bigg(1 - \sum_{i=0}^{2^{\rho-1}-2} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1 \cup \mathcal{B}_{[i]}^{-1}] \\ + \sum_{i=0}^{2^{\rho-1}-2} \frac{i}{2^\rho} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1 \cup \mathcal{B}_{[i]}^{-1}]\Bigg) 2n_u(1-\tau) + 9n_u\tau. \end{aligned} \tag{A.21}$$

According to Inequality (A.10), Inequality (A.14) and Condition (A.15), we have

$$\begin{aligned} \frac{\tau}{n_u} n_n^- &\leq \tau\left[1 - \sum_{i=0}^{2^{\rho-1}-2} \sum_{\mathcal{A}\in\mathbb{A}_i^1} (1-\tau)\left(1 - \frac{i+1}{2^\rho}\right) P[\boldsymbol{X} \in \mathcal{A}]\right] \\ &\leq \tau\left[1 - \frac{1}{2}(1-\tau) \sum_{i=0}^{2^{\rho-1}-2} \sum_{\mathcal{A}\in\mathbb{A}_i^1} P[\boldsymbol{X} \in \mathcal{A}]\right] \\ &\leq \tau\left[1 - \frac{1}{2}(1-\tau) \sum_{i=0}^{2^{\rho-1}-2} \left(P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1] - \tau^2\right)\right] \\ &= \tau\left\{1 - \frac{1}{2}(1-\tau)\left[\sum_{i=0}^{2^{\rho-1}-2} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1] - (2^{\rho-1}-1)\tau^2\right]\right\} \\ &\leq \tau\left[1 - \frac{1}{2}(1-\tau)\left(\sum_{i=0}^{2^{\rho-1}-2} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1] - \tau\right)\right] \leq \tau\left[1 - \frac{1}{2}(1-\tau)\left(D_- - 2\tau\right)\right], \end{aligned} \tag{A.22}$$

where $D_- = P[\boldsymbol{X} \in \boldsymbol{Z}_+] - P[\boldsymbol{X} \in S_{x[2^{\rho-1}-1]} \cap \boldsymbol{Z}_+]$.

Besides, according to Inequality (A.14) and Condition (A.16), the same as the content in [40] (Proof of Lemma 4), we have

$$n_3 \geq 2n_u(1-\tau)\left\{P[\boldsymbol{X} \in \mathcal{B}_{[2^{\rho-1}-1]}] - \left(\frac{1}{2} - \frac{1}{2^\rho}\right) P[\boldsymbol{X} \in \mathcal{B}_{[2^{\rho-1}-1]}]\right\} - 6n_u\tau. \tag{A.23}$$

Combining Inequality (A.21) and Inequality (A.23) with Inequality (A.9), Inequality (A.10), and Inequality (A.11), we can get

$$\begin{aligned} \frac{1}{n_u}\left((2+\tau)\left(n_u - n_1\right) - n_3\right) &\leq 2(1-\tau)\Bigg(1 - \sum_{i=0}^{2^{\rho-1}-1} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1 \cup \mathcal{B}_{[i]}^{-1}] \\ &\quad + \sum_{i=0}^{2^{\rho-1}-1} \frac{i}{2^\rho} P[\boldsymbol{X} \in \mathcal{B}_{[i]}^1 \cup \mathcal{B}_{[i]}^{-1}]\Bigg) + 15\tau \\ &\leq 2(1-\tau)\left(\tau + \sum_{i=0}^{2^{\rho-1}-1} \frac{i}{2^\rho} P\left[\boldsymbol{X} \in S_{x[i]}\right]\right) + 15\tau \leq 2(1-\tau)(\mathcal{R}_{\text{Bayes}} + 8.75\tau). \end{aligned} \tag{A.24}$$

Combining Inequality (A.21), Inequality (A.22) and Inequality (A.23), we can eventually obtain

$$\begin{aligned}\boldsymbol{\beta}_{[\mathrm{ap}]}^T\boldsymbol{\beta}_{[\mathrm{ap}]}+\frac{c'}{n_u}\sum_{l=1}^{n_u}\xi_{[\mathrm{ap}]l}&\le\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}+2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}\right.\\&\left.+(8.75+\frac{n_1^-}{2n_u(1-\tau)})\tau\right]\\&\le\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}+2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}+(8.75+\frac{n_n}{2n_u(1-\tau)})\tau\right]\\&\le\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}+2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}+(8.75+\frac{1-\frac{1}{2}(1-\tau)\left(D_--2\tau\right)}{2(1-\tau)})\tau\right].\end{aligned}\tag{A.25}$$

A.4.2. Lower bound of the inequality for contradiction

The way to construct the lower bound of the inequality for contradiction is quite similar to the corresponding content in [40] (proof of Lemma 6). One can easily achieve this by replacing the constraints condition of ξ_i in the proof of [40] with the constraints of ξ_i in Eq. (A.7). The same result as it in [40] (Lemma 6) can be obtained as

$$\begin{aligned}\frac{c'}{n_u}\sum_{l=1}^{n_u}\xi_{[\mathrm{ap}]l}&>(1-\tau)^2c'\left(2\mathcal{R}_{\mathrm{Bayes}}+\epsilon-11\tau\right)\\&=c'(1-\tau)\left(2\mathcal{R}_{\mathrm{Bayes}}+32\tau-11\tau-2\tau\mathcal{R}_{\mathrm{Bayes}}-\epsilon\tau+11\tau^2\right)\\&>c'(1-\tau)\left(2\mathcal{R}_{\mathrm{Bayes}}+19\tau\right).\end{aligned}\tag{A.26}$$

A.4.3. Construction of contradiction for the proof of Theorem 2

Combining Inequality (A.25) with Inequality (A.26) we can find

$$\begin{aligned}\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}&\ge\boldsymbol{\beta}_{[\mathrm{ap}]}^T\boldsymbol{\beta}_{[\mathrm{ap}]}+\frac{c'}{n_u}\sum_{l=1}^{n_u}\xi_{[\mathrm{ap}]l}-2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}\right.\\&\left.+(8.75+\frac{1-\frac{1}{2}(1-\tau)\left(D_--2\tau\right)}{2(1-\tau)})\tau\right]\\&\ge\frac{c'}{n_u}\sum_{l=1}^{n_u}\xi_{[\mathrm{ap}]l}-2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}\right.\\&\left.+(8.75+\frac{1-\frac{1}{2}(1-\tau)\left(D_--2\tau\right)}{2(1-\tau)})\tau\right]\\&>c'(1-\tau)\left(2\mathcal{R}_{\mathrm{Bayes}}+19\tau\right)-2c'(1-\tau)\left[\mathcal{R}_{\mathrm{Bayes}}\right.\\&\left.+(8.75+\frac{1-\frac{1}{2}(1-\tau)\left(D_--2\tau\right)}{2(1-\tau)})\tau\right]\\&=c'\tau\left[0.5-1.5\tau+\frac{1}{2}(1-\tau)\left(D_--2\tau\right)\right].\end{aligned}\tag{A.27}$$

It should be noted that

$$\tau\le\frac{1}{32}<0.2<\frac{1}{4}\min_{D_-\in[0,1]}\{5+D_--\sqrt{(5+D_-)^2-8(D_-+1)}\}.\tag{A.28}$$

Therefore, $0.5-1.5\tau+\frac{1}{2}(1-\tau)\left(D_--2\tau\right)>0$ holds for all $0<\tau=\frac{\epsilon}{32}\le\frac{1}{32}$. Then let

$$\begin{aligned}c^*&=\frac{\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}}{\tau\min_{D_-\in[0,1]}\{0.5-1.5\tau+\frac{1}{2}(1-\tau)\left(D_--2\tau\right)\}}\\&=\frac{2\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}}{\tau(1-5\tau+2\tau^2)}.\end{aligned}\tag{A.29}$$

For $c'\ge c^*$, we can obtain the contradiction according to Inequality (A.27) as

$$\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}>c^*\tau\left[0.5-1.5\tau+\frac{1}{2}(1-\tau)\left(D_--2\tau\right)\right]>\tilde{\boldsymbol{\beta}}^T\tilde{\boldsymbol{\beta}}.\tag{A.30}$$

Thus the assumption in Inequality (A.18) is false and we can draw a conclusion that, for $0<\epsilon=32\tau<1$ and $c'\ge c^*$,

$$\mathcal{R}\left(f_{\mathrm{ap}}^{*n_u}\right)-\mathcal{R}_{\mathrm{Bayes}}\le\epsilon$$

holds on the 'representative' dataset. Finally Theorem 2 is proved.

A.5. Step 4: Proof of Theorem 1

As the value of c' tends to infinity with n_u increasing, we can find n^* so that $c'\ge c^*$ when there is $n_u\ge n^*$. Then, according to Theorem 2, there is

$$P^{n_u}\left[\mathcal{R}\left(f_{\mathrm{ap}}^{*n_u}\right)-\mathcal{R}_{\mathrm{Bayes}}\le\epsilon\right]\ge1-2M_{n_u}e^{-\frac{\epsilon^6n_u}{2^{29}M_{n_u}^2}},\tag{A.31}$$

where $M_{n_u}=\frac{64}{\epsilon}\mathcal{N}\left((S_x,d_{\boldsymbol{\phi}}),\frac{\epsilon}{32\sqrt{c'}}\right)$. As $M_{n_u}^2\in\mathcal{O}\left(c'^{\alpha}\right)$ following the assumption on the covering number of $(S_x,d_{\boldsymbol{\phi}})$, $n_uM_{n_u}^{-2}$ tends to infinity with n_u increasing, and there is

$$P^{n_u}\left[\mathcal{R}\left(f_{\mathrm{ap}}^{*n_u}\right)-\mathcal{R}_{\mathrm{Bayes}}\le\epsilon\right]\to1.$$

As $\mathcal{R}\left(f_{\mathrm{uc}}^{*n_u}\right)\to\mathcal{R}\left(f_{\mathrm{ap}}^{*n_u}\right)$ with both n_p and n_u increasing, Theorem 1 is proved, i.e.,

$$P^{n_u}\left[\mathcal{R}\left(f_{\mathrm{uc}}^{*n_u}\right)-\mathcal{R}_{\mathrm{Bayes}}\le\epsilon\right]\to1.$$

Data availability

All datasets are publicly available. We have shared the link to the datasets in our paper.

References

[1] 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.

[2] H.S. Helm, A. Basu, A. Athreya, Y. Park, J.T. Vogelstein, C.E. Priebe, M. Winding, M. Zlatic, A. Cardona, P. Bourke, J. Larson, M. Abdin, P. Choudhury, W. Yang, C.W. White, Distance-based positive and unlabeled learning for ranking, Pattern Recognit. 134 (2023) 109085.

[3] Y. He, X. Li, M. Zhang, P. Fournier-Viger, J.Z. Huang, S. Salloum, A novel observation points-based positive-unlabeled learning algorithm, CAAI Trans. Intell. Technol. 8 (4) (2023) 1425–1443.

[4] 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.

[5] Z. Zhu, L. Wang, P. Zhao, C. Du, W. Zhang, H. Dong, B. Qiao, Q. Lin, S. Rajmohan, D. Zhang, Robust positive-unlabeled learning via noise negative sample self-correction, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 3663–3673.

[6] V. Sevetlidis, G. Pavlidis, S.G. Mouroutsos, A. Gasteratos, Dense-PU: Learning a density-based boundary for positive and unlabeled learning, IEEE Access (2024).

[7] C. Li, Y. Dai, L. Feng, X. Li, B. Wang, J. Ouyang, Positive and unlabeled learning with controlled probability boundary fence, in: Forty-First International Conference on Machine Learning, 2024.

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

[10] S. Kong, W. Shen, Y. Zheng, A. Zhang, J. Pu, J. Wang, False positive rate control for positive unlabeled learning, Neurocomputing 367 (2019) 13–19.

[11] B. Žunkovič, Positive unlabeled learning with tensor networks, Neurocomputing 552 (2023) 126556.

[12] N. Azizi, M. Ben Othmane, M. Hamouma, A. Siam, H. Haouassi, M. Ledmi, A. Hamdi-Cherif, BiCSA-PUL: binary crow search algorithm for enhancing positive and unlabeled learning, Int. J. Inf. Technol. (2024) 1–15.

[13] 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.

[14] T. Ke, H. Lv, M. Sun, L. Zhang, A biased least squares support vector machine based on Mahalanobis distance for PU learning, Phys. A 509 (2018) 422–438.

[15] 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.
[16] M.C. Du Plessis, G. Niu, M. Sugiyama, Analysis of learning from positive and unlabeled data, Adv. Neural Inf. Process. Syst. 27 (2014).
[17] R. Kiryo, G. Niu, M.C. Du Plessis, M. Sugiyama, Positive-unlabeled learning with non-negative risk estimator, Adv. Neural Inf. Process. Syst. 30 (2017).
[18] 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.
[19] X. Wang, X. Yang, R. Zhu, J.-H. Xue, PUAL: A classifier on trifurcate positive-unlabelled data, Neurocomputing 637 (2025) 130080.
[20] J. Wilton, A. Koay, R. Ko, M. Xu, N. Ye, Positive-unlabeled learning using random forests via recursive greedy risk minimization, Adv. Neural Inf. Process. Syst. 35 (2022) 24060–24071.
[21] C. Ortega Vázquez, S. vanden Broucke, J. De Weerdt, Hellinger distance decision trees for PU learning in imbalanced data sets, Mach. Learn. 113 (7) (2024) 4547–4578.
[22] Y. Zhao, M. Zhang, C. Zhang, W. Chen, N. Ye, M. Xu, A boosting framework for positive-unlabeled learning, Stat. Comput. 35 (1) (2025) 2.
[23] 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.
[24] A. Kumagai, T. Iwata, H. Takahashi, T. Nishiyama, Y. Fujiwara, AUC maximization under positive distribution shift, Adv. Neural Inf. Process. Syst. 37 (2024) 36071–36096.
[25] X. Chen, C. Gong, J. Yang, Cost-sensitive positive and unlabeled learning, Inform. Sci. 558 (2021) 229–245.
[26] Y. Liu, J. Zhao, Y. Xu, Robust and unbiased positive and unlabeled learning, Knowl.-Based Syst. 277 (2023) 110819.
[27] X. Wang, H. Chen, T. Guo, Y. Wang, PUe: Biased positive-unlabeled learning enhancement by causal inference, Adv. Neural Inf. Process. Syst. 36 (2023) 19783–19798.
[28] P. Zhao, J. Deng, X. Cheng, Soft label PU learning, 2024, arXiv preprint arXiv:2405.01990.
[29] H. Gu, H.X. Tae, C.S. Chan, L. Fan, A few-shot label unlearning in vertical federated learning, 2024, arXiv preprint arXiv:2410.10922.
[30] Z. Yuan, K. Zhang, T. Huang, Positive label is all you need for multi-label classification, in: IEEE International Conference on Multimedia and Expo, ICME, IEEE, 2024, pp. 1–6.
[31] N. Natarajan, I.S. Dhillon, P.K. Ravikumar, A. Tewari, Learning with noisy labels, Adv. Neural Inf. Process. Syst. 26 (2013).
[32] D. Gabay, B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, Comput. Math. Appl. 2 (1) (1976) 17–40.
[33] Y. Wang, W. Yin, J. Zeng, Global convergence of ADMM in nonconvex nonsmooth optimization, J. Sci. Comput. 78 (2019) 29–63.
[34] S. Jain, M. White, M.W. Trosset, P. Radivojac, Nonparametric semi-supervised learning of class proportions, 2016, arXiv preprint arXiv:1601.01944.
[35] M. Christoffel, G. Niu, M. Sugiyama, Class-prior estimation for learning from positive and unlabeled data, in: Asian Conference on Machine Learning, PMLR, 2016, pp. 221–236.
[36] J. Bekker, J. Davis, Estimating the class prior in positive and unlabeled data through decision tree induction, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018, 1.
[37] I. Steinwart, Consistency of support vector machines and other regularized kernel classifiers, IEEE Trans. Inform. Theory 51 (1) (2005) 128–142.
[38] L. Deng, The MINST database of handwritten digit images for machine learning research, IEEE Signal Process. Mag. 29 (6) (2012) 141–142.
[39] H. Zhao, X. Wang, J. Li, Y. Zhong, Class prior-free positive-unlabeled learning with taylor variational loss for hyperspectral remote sensing imagery, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 16827–16836.
[40] I. Steinwart, Support vector machines are universally consistent, J. Complexity 18 (3) (2002) 768–791.

Xiaoke Wang received the Ph.D. degree in Machine Learning from University College London in 2024, focusing on positive-unlabelled learning. He also holds an M.Sc. in Statistics from University College London and a Bachelor's degree in Statistics from Zhongnan University of Economics and Law. His research interests include machine learning and statistical data analysis. He has served as a reviewer for IEEE Transactions on Circuits and Systems for Video Technology, IEEE Transactions on Cybernetics, and IEEE Transactions on Neural Networks and Learning Systems.

Rui Zhu received the Ph.D. degree in statistics from University College London in 2017. She is a Senior Lecturer in the Faculty of Actuarial Science and Insurance, City, University of London. Her research interests include machine learning and its applications in image quality assessment, hyperspectral image analysis and actuarial science. She is an Associate Editor of Neurocomputing, the IEEE Transactions on Circuits and Systems for Video Technology and the IEEE Transactions on Neural Networks and Learning Systems.

Jing-Hao Xue received the Dr.Eng. degree in signal and information processing from Tsinghua University in 1998 and the Ph.D. degree in statistics from the University of Glasgow in 2008. He is a Professor in the Department of Statistical Science at University College London. His research interests include statistical pattern recognition, machine learning and computer vision. He is an Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology, the IEEE Transactions on Cybernetics, and the IEEE Transactions on Neural Networks and Learning Systems.