Neural Networks 157 (2023) 125–135

Contents lists available at ScienceDirect

Neural Networks

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

Inverse free reduced universum twin support vector machine for imbalanced data classification

Hossein Moosaei [a,b,*], M.A. Ganaie [c,d], Milan Hladík [b], M. Tanveer [c]

[a] Department of Informatics, Faculty of Science, Jan Evangelista Purkyně University, Ústí nad Labem, Czech Republic
[b] Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic
[c] Department of Mathematics, Indian Institute of Technology Indore, Simrol, Indore, 453552, India
[d] Department of Robotics, University of Michigan, Ann Arbor, MI, 48109, USA

ARTICLE INFO

Article history:
Received 8 March 2022
Revised and accepted 4 October 2022
Available online 15 October 2022

Keywords:
Universum
Class-imbalanced
Twin support vector machine
Universum twin support vector machine
Rectangular kernel
Reduced universum twin support vector machine

ABSTRACT

Imbalanced datasets are prominent in real-world problems. In such problems, the data samples in one class are significantly higher than in the other classes, even though the other classes might be more important. The standard classification algorithms may classify all the data into the majority class, and this is a significant drawback of most standard learning algorithms, so imbalanced datasets need to be handled carefully. One of the traditional algorithms, twin support vector machines (TSVM), performed well on balanced data classification but poorly on imbalanced datasets classification. In order to improve the TSVM algorithm's classification ability for imbalanced datasets, recently, driven by the universum twin support vector machine (UTSVM), a reduced universum twin support vector machine for class imbalance learning (RUTSVM) was proposed. The dual problem and finding classifiers involve matrix inverse computation, which is one of RUTSVM's key drawbacks. In this paper, we improve the RUTSVM and propose an improved reduced universum twin support vector machine for class imbalance learning (IRUTSVM). We offer alternative Lagrangian functions to tackle the primal problems of RUTSVM in the suggested IRUTSVM approach by inserting one of the terms in the objective function into the constraints. As a result, we obtain new dual formulation for each optimization problem so that we need not compute inverse matrices neither in the training process nor in finding the classifiers. Moreover, the smaller size of the rectangular kernel matrices is used to reduce the computational time. Extensive testing is carried out on a variety of synthetic and real-world imbalanced datasets, and the findings show that the IRUTSVM algorithm outperforms the TSVM, UTSVM, and RUTSVM algorithms in terms of generalization performance.

1. Introduction

Support vector machine (SVM), which is rooted in the idea of structural risk minimization (SRM), is one of the famous approaches for regression and pattern classification (Cervantes, Garcia-Lamont, Rodríguez-Mazahua, & Lopez, 2020; Vapnik, 2013; Vapnik & Chervonenkis, 1974). This method has been used in many applications such as medicine (heart disease, lung cancer, or colon tumor prediction) text categorization, computational biology, bioinformatics, image classification, real-life datasets, etc. (Arabasadi, Alizadehsani, Roshanzamir, Moosaei, & Yarifard, 2017; Bazikar, Ketabchi, & Moosaei, 2020; Cai, Ricardo, Jen, & Chou, 2004; Javadi, Moosaei, & Ciuonzo, 2019; Ketabchi, Moosaei, Razzaghi, & Pardalos, 2019; Noble, 2004; Tong & Koller, 2001; Wang, Wang, & Bu, 2011). SVM seeks two parallel hyperplanes such that the distance between them is maximal. This strategy leads to solving a constrained quadratic programming problem (QPP).

Since all data samples are visible in the constraints of the QPP in the SVM, the method has a high computational cost. To decrease SVM computational cost, the generalized eigenvalue proximal SVMs (GEPSVMs) Mangasarian and Wild (2005) was proposed. The GEPSVM seeks two nonparallel hyperplanes in a way that each hyperplane is close to one of the classes and farthest from the other class. By similar idea, Jayadeva, Khemchandani, and Chandra (2007) suggested a twin support vector machine (TSVM). The formulation of TSVM is similar to SVM and different from that of GEPSVMs. For finding two nonparallel hyperplanes, TSVM solves two QPPs of smaller size rather than one larger QPP as in SVM; so this method is faster than the SVM (Jayadeva et al., 2007). Many researchers investigated TSVM and introduced improvements to it, e.g., twin bounded support vector machine

* Corresponding author at: Department of Informatics, Faculty of Science, Jan Evangelista Purkyně University, Ústí nad Labem, Czech Republic.
E-mail addresses: hossein.moosaei@ujep.cz (H. Moosaei), phd1901141006@iiti.ac.in, mudasirg@umich.edu (M.A. Ganaie), hladik@kam.mff.cuni.cz (M. Hladík), mtanveer@iiti.ac.in (M. Tanveer).

https://doi.org/10.1016/j.neunet.2022.10.003
0893-6080/

(TBSVM) proposed by Shao, Zhang, Wang, and Deng (2011). The main advantage of TBSVM compared to TSVM is that the principle of structural risk minimization is carried out by introducing the regularization term. Other extensions to the TSVM can be found in Chen, Shao, Li, Liu, Wang, and Deng (2020), Ganaie, Tanveer, and Lin (2022), Huang, Wei, and Zhou (2018), Kumar and Gopal (2009), Ma, Yang, and Sun (2020), Moosaei, Ketabchi, Razzaghi, and Tanveer (2021), Tanveer, Sharma, and Suganthan (2019), Tanveer, Tiwari, Choudhary, and Ganaie (2021). A recent evaluation of the TSVM based models (Tanveer, Gautam, & Suganthan, 2019) shows that robust energy based least squares TSVM (RELSTSVM) (Tanveer, Khan, & Ho, 2016) is the best performing model among the TSVM based models. For comprehensive review of the TSVM, we refer the interested readers to Tanveer, Rajani, Rastogi, Shao, and Ganaie (2022). Universum data comprises a collection of a sample that does not belong to any of the concerned classes and provides some information about the distribution of data. In an effort to include previous information about data distribution in the classifier construction, Weston, Collobert, Sinz, Bottou, and Vapnik (2006) introduced the universum support vector machine (USVM). Qi, Tian, and Shi (2012) proposed the universum twin support vector machine (UTSVM) to decrease USVM's computational time. Following this approach, some researchers further focused on improvements of UTSVM leading (Kumar & Gupta, 2021; Richhariya, Sharma, & Tanveer, 2018; Richhariya, Tanveer, & Initiative, 2020; Xu, Chen, & Li, 2016). Many different kinds of applications make use of classification algorithm that are based on the universum. For instance, the classification algorithms with universum approach have been studied in EEG signal classification (Gupta, Sarma, Mishra, & Prasad, 2019; Richhariya & Tanveer, 2018), Alzheimer's disease's diagnosis (Richhariya, Tanveer, Rashid, & Initiative, 2020), and facial expression recognition (Richhariya & Gupta, 2019).

The problem of class imbalance, frequently occurring in real-life applications, is one the challenging problems in machine learning and data mining (Chawla, Japkowicz, & Kotcz, 2004). This problem occurs when the training data is not distributed between the classes evenly so that the number of training data samples in one class is significantly larger than the number of training data in the other class. For classifying imbalanced data, the standard methods such as SVM, TSVM, USVM, UTSVM, and others become biased towards the majority class, and the minority class's data points get misclassified because they cannot contribute much to the learning process. To tackle the class imbalance problem, many methods have been proposed by scholars. Some of them improved the classification algorithms such as the ensemble learning (Ganaie & Tanveer, 2020; Ganaie, Tanveer, & Suganthan, 2020; Wang, Xing, Li, Hua, Dong, & Pedrycz, 2014), SVM (Tang, Zhang, Chawla, & Krasser, 2008), the efficient weighted Lagrangian TSVM (Shao, Chen, Zhang, Wang, & Deng, 2014), and so on. In order to use universum learning for the data of class imbalance, Richhariya and Tanveer (2020) recently proposed an algorithm called reduced universum twin support vector machine for class imbalance learning (RUTSVM). There are two drawbacks of RUTSVM. First, one of the most distinguishing features of RUTSVM, as well as all TSVM-based models, is that they all use matrix inverse operations. Computing the inverse of a matrix is computationally expensive (Wang, He, Chen, & Yeung, 2005) and also the matrix may not be invertible. In RUTSVM not only we have to compute the inverse matrices before the training process, but we also have to calculate the inverse matrices after the training for finding the hyperplanes. Also a small positive value for discovering classifiers should be added to prevent singular matrices, which implies classifiers are not very precise. Second, if we use the linear kernel for nonlinear RUTSVM, we will not face an equivalent problem to the linear case.

To address these issues, this paper introduces a technique to improve the reduced universum twin support vector machine for class imbalance learning and named it as IRUTSVM. The proposed IRUTSVM includes putting one of the terms from objective function into the constraints, we give new equivalent formulations for primal problems. There are also a variety of Lagrangian functions and, as a result, a variety of dual problems. It is worth mentioning that in dual problems and hyperplanes formulations of the proposed IRUTSVM, there is no need to compute inverse matrices. Additionally, for nonlinear IRUTSVM, different kernels are constructed for the primal problems so that it is superior to nonlinear RUTSVM. As a result, the drawbacks of RUTSVM are efficiently handled by the proposed IRUTSVM model. Benchmarks and simulated datasets show that the proposed IRUTSVM method is successful and feasible in terms of generalization performance.

This paper makes several contributions to the literature including:

- IRUTSVM reformulates the primal form of RUTSVM to make it theoretically more stable.
- IRUTSVM minimizes the structural risk which is the marrow of statistical learning. Implementation of the structural risk minimization (SRM) principle overcomes the overfitting issues, resulting in better generalization performance.
- By remodeling the primal problems of RUTSVM, we present various Lagrangian functions for them and then we have different dual problems. This allows us to avoid complications of matrix inverse computation in the dual problems and classifiers while other given baseline models need to calculate the matrix inverse which is infeasible in real-world scenarios (as the matrix may not be invertible). Also, singularity issues are not present in the proposed model.
- Unlike the given baseline models which employ kernel-generated surfaces for the nonlinear case, the proposed IRUTSVM incorporates the kernel trick directly into its dual problem.
- The proposed IRUTSVM method has higher efficiency than RUTSVM in the aspect of accuracy, therefore it is suitable for imbalanced data classification.

This paper is structured as follows: Section 2 offers a brief review of TSVM, UTSVM, and RUTSVM. Section 3 discusses the presented method IRUTSVM in both nonlinear and linear cases. Section 4 displays numerical experiments on multiple UCI real world and NDC datasets. Conclusions are drawn in Section 5.

Notations. The n-dimensional real space is denoted by R^n. The A^T and $\|\cdot\|$ stand for the transpose of a matrix A and the Euclidean norm, respectively. We have denoted the scalar product of two vectors x and y in the n-dimensional real space R^n by $x^T y$. A column vector of one of arbitrary dimension is displayed by e, and I_n stands for the identity matrix of size $n \times n$. For $A \in R^{m\times n}$ and $B \in R^{n\times l}$, the kernel $K(A; B)$ represents an arbitrary function that provides a mapping of $R^{m\times n} \times R^{n\times l}$ into $R^{m\times l}$. In particular, if x and y are column vectors in R^n, then $K(x^T; y)$ is a real number, $K(x^T; A^T)$ is a row vector in R^m and $K(A; A^T)$ is an $m \times m$ matrix.

Throughout the paper, the data points in class $+1$ are represented by matrix $A \in R^{m_1\times n}$ and the class -1's data points are shown by matrix $B \in R^{m_2\times n}$. Universum data are represented by matrix U. The imbalance ratio (IR) formula is as follows:

$$IR = \frac{Number\ of\ majority\ class\ samples}{Number\ of\ minority\ class\ samples}.$$

2. Related works

In this section, a brief review of TSVM, UTSVM, and RUTSVM is presented.

2.1. Twin support vector machine

Twin support vector machine (TSVM), developed by Jayadeva et al. (2007), identifies two nonparallel hyperplanes for binary classification in a way that each hyperplane is as much as possible close to one of the two classes and farthest from the other class. The main idea of TSVM was inspired by GEPSVM (Mangasarian & Wild, 2005). Notice that TSVM has a formulation similar to SVM formulation except that only some datasets are shown in either problem's constraints simultaneously. This makes TSVM faster than standard SVM (Jayadeva et al., 2007).

TSVM finds two nonparallel hyperplanes as described below:

$$f_1(x) = w_1^T x + b_1 \quad \text{and} \quad f_2(x) = w_2^T x + b_2, \tag{1}$$

where $w_1 \in R^n$, $w_2 \in R^n$, $b_1 \in R$ and $b_2 \in R$.

The TSVM classifiers are achieved by finding a solution to the pair of quadratic programming problems below:

$$\begin{aligned} \min_{w_1, b_1, q_1} \quad & \frac{1}{2}\|Aw_1 + e_1 b_1\|^2 + c_1 e_2^T q_1, \\ \text{s.t.} \quad & -(Bw_1 + e_2 b_1) + q_1 \geq e_2, \quad q_1 \geq 0, \end{aligned}$$

$$\begin{aligned} \min_{w_2, b_2, q_2} \quad & \frac{1}{2}\|Bw_2 + e_2 b_2\|^2 + c_2 e_1^T q_2, \\ \text{s.t.} \quad & (Aw_2 + e_1 b_2) + q_2 \geq e_1, \quad q_2 \geq 0, \end{aligned}$$

where $c_1, c_2 > 0$ are penalty parameters, e_1, e_2 denote vectors related to ones of suitable dimensions, and q_1 and q_2 are slack vectors.

By using the KKT conditions, we derive the Wolfe dual formulations of (2) and (2), respectively, as follows:

$$\max_{\alpha} \; e_2^T \alpha - \frac{1}{2}\alpha^T G (H^T H)^{-1} G^T \alpha \quad \text{s.t.} \;\; 0 \leq \alpha \leq c_1. \tag{2}$$

and

$$\max_{\gamma} \; e_1^T \gamma - \frac{1}{2}\gamma^T H (G^T G)^{-1} H^T \gamma \quad \text{s.t.} \;\; 0 \leq \gamma \leq c_2, \tag{3}$$

where α and γ are the Lagrangian coefficients, $H = [A \; e_1]$ and $G = [B \; e_2]$. The separating hyperplanes can be obtained from the solution of (2) and (3) by evaluating

$$[w_1^T, \; b_1]^T = -(H^T H)^{-1} G^T \alpha,$$

and

$$[w_2^T, \; b_2]^T = (G^T G)^{-1} H^T \gamma,$$

respectively.

To avoid the possible ill-conditioning, when $G^T G$ or $H^T H$ are (nearly) singular, the inverse matrices $(G^T G)^{-1}$ and $(H^T H)^{-1}$ are approximately substituted by $(G^T G + \delta I_{n+1})^{-1}$ and $(H^T H + \delta I_{n+1})^{-1}$, where δ is a small positive scalar.

2.2. Universum twin support vector machine

The universum twin support vector machine (UTSVM) was presented by Qi et al. (2012). They incorporated universum data into TSVM and improved generalization by drawing the prior knowledge which was embedded in universum. UTSVM solves two smaller sized QPPs rather than one large QPP in USVM, so that UTSVM will be faster than USVM. Notice that UTSVM also uses the ϵ-insensitive loss function to locate universum data in a nonparallel insensitive loss tube. The nonlinear UTSVM can solve the pair of QPPs below

$$\begin{aligned} \min_{w_1, b_1, q_1, \phi_1} \quad & \frac{1}{2}\|K(A, D^T)w_1 + e_1 b_1\|^2 + c_1 e_2^T q_1 + c_u e_u^T \phi_1, \\ \text{s.t.} \quad & -(K(B, D^T)w_1 + e_2 b_1) + q_1 \geq e_2, \\ & (K(U, D^T)w_1 + e_u b_1) + \phi_1 \geq (-1+\epsilon)e_u, \\ & q_1, \phi_1 \geq 0, \end{aligned} \tag{4}$$

$$\begin{aligned} \min_{w_2, b_2, q_2, \phi_2} \quad & \frac{1}{2}\|K(B, D^T)w_2 + e_2 b_2\|^2 + c_2 e_1^T q_2 + c_u e_u^T \phi_2, \\ \text{s.t.} \quad & (K(A, D^T)w_2 + e_1 b_2) + q_2 \geq e_1, \\ & -(K(U, D^T)w_2 + e_u b_2) + \phi_2 \geq (-1+\epsilon)e_u, \\ & q_2, \phi_2 \geq 0, \end{aligned} \tag{5}$$

where c_1, $c_2 > 0$ and $c_u > 0$ are parameters, e_1, e_2 and e_u are vectors of ones of suitable dimensions, q_1, q_2, ϕ_1 and ϕ_2 are slack vectors, and $D^T = [A^T, B^T]$.

By applying the KKT conditions, the Wolfe dual formulations of (4) and (5) can be attained, respectively as follows:

$$\begin{aligned} \max_{\alpha_1, \mu_1} \quad & e_2^T \alpha_1 - \frac{1}{2}(\alpha_1^T G - \mu_1^T O)(H^T H)^{-1}(G^T \alpha_1 - O^T \mu_1) \\ & + (\epsilon - 1)e_u^T \mu_1, \\ \text{s.t.} \quad & 0 \leq \alpha_1 \leq c_1, \quad 0 \leq \mu_1 \leq c_u, \end{aligned} \tag{6}$$

$$\begin{aligned} \max_{\alpha_2, \mu_2} \quad & e_1^T \alpha_2 - \frac{1}{2}(\alpha_2^T H - \mu_2^T O)(G^T G)^{-1}(H^T \alpha_2 - O^T \mu_2) \\ & + (\epsilon - 1)e_u^T \mu_2, \\ \text{s.t.} \quad & 0 \leq \alpha_2 \leq c_2, \quad 0 \leq \mu_2 \leq c_u, \end{aligned} \tag{7}$$

where α_1, α_2, μ_1 and μ_2 are the Lagrangian coefficients, $H = [K(A, D^T) \; e_1]$, $G = [K(B, D^T) \; e_2]$, and $O = [K(U, D^T) \; e_u]$. The separating hyperplanes are determined from the solution of (6) and (7) by evaluating

$$[w_1^T, \; b_1]^T = -(H^T H)^{-1}(G^T \alpha_1 - O^T \mu_1),$$

and

$$[w_2^T, \; b_2]^T = (G^T G)^{-1}(H^T \alpha_2 - O^T \mu_2).$$

To avoid the possible ill-conditioning, when $G^T G$ or $H^T H$ are (nearly) singular, the inverse matrices $(G^T G)^{-1}$ and $(H^T H)^{-1}$ are approximately substituted by $(G^T G + \delta I)^{-1}$ and $(H^T H + \delta I)^{-1}$, where δ is a small positive scalar.

A new data point x is allocated to class $i \in \{+1, -1\}$ by using the following decision rule:

$$\text{class } i = \arg\min_{i=1,2} \frac{|K(x^T, D^T)w_i + b_i|}{\|w_i\|}.$$

2.3. Reduced universum twin support vector machine for class imbalance learning (RUTSVM)

In rest of the paper, while maintaining generality, we suppose that the positive class is the minority class and for creating the balance problem, the negative class's random undersampling is used to construct a hyperplane of the positive class. Also, matrix $B^* \in R^{m_1 \times n}$ is a randomly chosen reduced data sample from the negative class. The universum data points are associated with matrix $U \in R^{r \times n}$, where $r = m_2 - m_1$, and we also denote a random subset of U by matrix $U^* \in g \times n$, where $g = \lceil \frac{m_1}{2} \rceil$. Let $D^T = [A^T, B^{*T}]$.

The primal problem of nonlinear RUTSVM for finding the first hyperplane is as follows:

$$\begin{aligned} \min_{w_1, b_1, \xi, \psi} \quad & \frac{1}{2}\|K(A, D^T)w_1 + e_1 b_1\|^2 + C_1 e_1^T \xi + C_u e_g^T \psi \\ \text{s.t.} \quad & -(K(B^*, D^T)w_1 + e_1 b_1) + \xi \geq e_1, \end{aligned} \tag{8}$$

$$(K(U^*, D^T)w_1 + e_g b_1) + \psi \geq (-1+\varepsilon)e_g,$$
$$\xi, \psi \geq 0.$$

The primal problem of nonlinear RUTSVM for the second hyperplane reads

$$\begin{aligned} \min_{w_2, b_2, \xi, \psi} \quad & \frac{1}{2}\|K(B, D^T)w_2 + e_2 b_2\|^2 + C_2 e_1^T \xi + C_u e_r^T \psi \\ \text{s.t.} \quad & (K(A, D^T)w_2 + e_1 b_2) + \xi \geq e_1, \\ & (K(U, D^T)w_2 + e_r b_2) + \psi \geq (1-\varepsilon)e_r, \\ & \xi, \psi \geq 0. \end{aligned} \tag{9}$$

After applying the KKT conditions, the Wolfe duals of objective functions (8) and (9) are given as follows:

$$\begin{aligned} \max_{\alpha_1, \mu_1} \quad & e_1^T\alpha_1 - \frac{1}{2}(\alpha_1^T P^* - \mu_1^T O^*)(S^T S)^{-1}(P^{*T}\alpha_1 - O^{*T}\mu_1) \\ & + (\epsilon - 1)e_g^T\mu_1 \\ \text{s.t.} \quad & 0 \leq \alpha_1 \leq C_1, \quad 0 \leq \mu_1 \leq C_u \end{aligned} \tag{10}$$

and

$$\begin{aligned} \max_{\alpha_2, \mu_2} \quad & e_1^T\alpha_2 - \frac{1}{2}(\alpha_2^T S + \mu_2^T O)(P^T P)^{-1}(S^T\alpha_2 + O^T\mu_2) + (1-\epsilon)e_r^T\mu_2 \\ \text{s.t.} \quad & 0 \leq \alpha_2 \leq C_2, \quad 0 \leq \mu_2 \leq C_u, \end{aligned} \tag{11}$$

where $S = [K(A, D^T), e_1]$, $P^* = [K(B^*, D^T), e_1]$, $P = [K(B, D^T), e_2]$, $O^* = [K(U^*, D^T), e_g]$, $O = [K(U, D^T), e_r]$ and α_i, μ_i are the Lagrange multipliers, with $i = 1, 2$.

The optimal hyperplanes are given as :

$$[w_1^T,\ b_1]^T = -(S^T S + \delta I)^{-1}(P^{*T}\alpha_1 - O^{*T}\mu_1), \tag{12}$$
$$[w_2^T,\ b_2]^T = (P^T P + \delta I)^{-1}(S^T\alpha_2 + O^T\mu_2) \tag{13}$$

where, δ is a small positive number.

For new data sample $x \in R^n$, the class label is assigned as

$$class\ i = \arg\min_{i=1,2} \frac{|K(x^T, D^T)w_i + b_i|}{\|w_i\|}.$$

3. Improvement on reduced universum twin support vector machine for imbalanced classes

This section proposes an improvement of the RUTSVM (Richhariya & Tanveer, 2020) and call our method IRUTSVM. Most TWSVM-based techniques have two significant flaws: they calculate inverse matrices, which can result in unsatisfactory solutions in some cases and make them inappropriate for analysis, and they use kernel generated surfaces for the nonlinear situation. The proposed IRUTSVM which is a new version of TBSVM classifies the data without facing the mentioned flaws. Theoretically, we reduce the dual problems. In our calculation we avoid computing inverse matrices, that appear in classical methods.

3.1. Linear IRUTSVM

The linear IRUTSVM finds a pair of nonparallel hyperplanes as follows:

$$w_1^T x + b_1 = 0, \quad \text{and} \quad w_2^T x + b_2 = 0, \tag{14}$$

where $w_1 \in R^n$, $w_2 \in R^n$, $b_1 \in R$ and $b_2 \in R$.

The primal problem of linear IRUTSVM for finding the first hyperplane is as follows:

$$\begin{aligned} \min_{w_1, b_1, \xi, \psi, t} \quad & \frac{1}{2}\|t\|^2 + \frac{C_2}{2}(\|w_1\|^2 + b_1^2) + C_1 e_1^T\xi + C_u e_g^T\psi \\ \text{s.t.} \quad & Aw_1 + e_1 b_1 = t, \\ & -(B^* w_1 + e_1 b_1) + \xi \geq e_1, \\ & (U^* w_1 + e_g b_1) + \psi \geq (-1+\varepsilon)e_g, \\ & \xi, \psi \geq 0, \end{aligned} \tag{15}$$

where C_1, C_2, C_u are non negative penalty parameters, ξ, ψ represent the slack variables, and e_u, e_1, e_2 are vectors of ones of appropriate dimensions.

The Lagrangian function of problem (15) can be written as follows:

$$\begin{aligned} L(\theta) = & \frac{1}{2}\|t\|^2 + \frac{C_2}{2}(\|w_1\|^2 + b_1^2) + C_1 e_1^T\xi + C_u e_g^T\psi \\ & + \lambda^T(Aw_1 + e_1 b_1 - t) - \alpha^T(-(B^* w_1 + e_1 b_1) + \xi - e_1) \\ & - \beta^T\left((U^* w_1 + e_g b_1) + \psi + (1-\varepsilon)e_g\right) - p^T\xi - q^T\psi, \end{aligned} \tag{16}$$

where $\theta = (w_1, b_1, t, \xi, \psi, \lambda, \alpha, \beta, p, q)$ and $w_1 \in R^n$, $b_1 \in R$, $t \in R^{m_1}$, $\xi \in R^{m_1}$, $\psi \in R^{m_g}$, $\lambda \in R^{m_1}$, $\alpha \in R^{m_1}$, $\beta \in R^{m_g}$, $p \in R^{m_1}$ and $q \in R^{m_g}$. We note that λ, α, β, p and q are Lagrangian multipliers.

According to the KKT conditions, the following conditions are satisfied:

$$\frac{dL}{dw_1} = C_2 w_1 + A^T\lambda + B^{*T}\alpha - U^{*T}\beta = 0 \tag{17}$$
$$\frac{dL}{db_1} = C_2 b_1 + e_1^T\lambda + e_1^T\alpha - e_g^T\beta = 0 \tag{18}$$
$$\frac{dL}{dt} = t - \lambda = 0 \tag{19}$$
$$\frac{dL}{d\xi} = C_1 e_1 - \alpha - p = 0 \tag{20}$$
$$\frac{dL}{d\psi} = C_u e_g - \beta - q = 0 \tag{21}$$
$$\frac{dL}{d\lambda} = Aw_1 + e_1 b_1 - t = 0 \tag{22}$$
$$\frac{dL}{d\alpha} = -(-(B^* w_1 + e_1 b_1) + \xi - e_1) \leq 0 \tag{23}$$
$$\alpha^T\frac{dL}{d\alpha} = \alpha^T(-(B^* w_1 + e_1 b_1) + \xi - e_1) = 0 \tag{24}$$
$$\frac{dL}{d\beta} = -((U^* w_1 + e_g b_1) + \psi + (1-\varepsilon)e_g) \leq 0 \tag{25}$$
$$\beta^T\frac{dL}{d\beta} = \beta^T((U^* w_1 + e_g b_1) + \psi + (1-\varepsilon)e_g) = 0 \tag{26}$$
$$\frac{dL}{dp} = -\xi \leq 0 \tag{27}$$
$$p^T\frac{dL}{dp} = p^T\xi = 0 \tag{28}$$
$$\frac{dL}{dq} = -\psi \leq 0 \tag{29}$$
$$q^T\frac{dL}{dq} = q^T\psi = 0 \tag{30}$$
$$\alpha, \beta, p, q \geq 0 \tag{31}$$

From (17), (18), (19), we can obtain:

$$w_1 = -\frac{1}{C_2}\left(A^T\lambda + B^{*T}\alpha - U^{*T}\beta\right) \tag{32}$$
$$b_1 = -\frac{1}{C_2}\left({e_1}^T\lambda + {e_1}^T\alpha - {e_g}^T\beta\right) \tag{33}$$
$$t = \lambda \tag{34}$$

From (20), (21) and (31) we have $0 \le \alpha \le C_1 e_1$ and $0 \le \beta \le C_u e_g$. By substituting (32), (33) and (34) into (16), the Wolfe dual problem of the primal problem (15) can be expressed as follows:

$$\begin{aligned}\max_{\lambda,\alpha,\beta}\ & -\frac{1}{2}\begin{bmatrix}\lambda^T & \alpha^T & \beta^T\end{bmatrix} Q \begin{bmatrix}\lambda\\ \alpha\\ \beta\end{bmatrix} \\ & + C_2\begin{bmatrix}0 & e_1^T & (\varepsilon-1)e_g^T\end{bmatrix}\begin{bmatrix}\lambda\\ \alpha\\ \beta\end{bmatrix}, \\ \text{s.t. } & 0 \le \alpha \le C_1 e_1, \quad 0 \le \beta \le C_u e_g, \end{aligned} \tag{35}$$

where

$$Q = \begin{bmatrix} AA^T + C_2 I & AB^{*T} & -AU^{*T} \\ B^*A^T & B^*B^{*T} & -B^*U^{*T} \\ -U^*A^T & -U^*B^{*T} & U^*U^{*T} \end{bmatrix} + \begin{bmatrix} E & E & -E \\ E & E & -E \\ -E & -E & E \end{bmatrix},$$

and E is a matrix with all entries equal to 1.

Problem (35) can be stated as follows:

$$\begin{aligned}\max_{\lambda,\alpha,\beta}\ & \frac{1}{2}S^T Q S - c_2 r^T S \\ \text{s.t. } & 0 \le \alpha \le C_1 e_1, \\ & 0 \le \beta \le C_u e_g, \end{aligned} \tag{36}$$

where

$$Q = Q^T, \quad S = \begin{bmatrix}\lambda\\ \alpha\\ \beta\end{bmatrix}, \quad r = \begin{bmatrix}0\\ e_1\\ (\varepsilon-1)e_g\end{bmatrix},$$

and Q is a symmetric positive definite matrix.

The primal problem of IRUTSVM for the second hyperplane is as follows:

$$\begin{aligned}\min_{w_2,b_2,\xi,\psi,t}\ & \frac{1}{2}\|t\|^2 + \frac{C_4}{2}(\|w_2\|^2 + b_2^2) + C_3 e_1^T \xi + C_u e_r^T \psi \\ \text{s.t. } & Bw_2 + e_2 b_2 = t, \\ & (Aw_2 + e_1 b_2) + \xi \ge e_1 \\ & (Uw_2 + e_r b_2) + \psi \ge (1-\varepsilon)e_r \\ & \xi, \psi \ge 0, \end{aligned} \tag{37}$$

where C_3, C_4, C_u are nonnegative penalty parameters, ξ, ψ denote the slack variables, and e_r, e_1, e_2 are vectors of ones of appropriate dimensions.

Similarly to the problem (15), the Wolfe dual problem for the primal problem (37) is obtained as follows:

$$\begin{aligned}\max_{\lambda,\alpha,\beta}\ & \frac{1}{2}S^T Q S - c_4 r^T S \\ \text{s.t. } & 0 \le \alpha \le C_3 e_1, \\ & 0 \le \beta \le C_u e_r, \end{aligned} \tag{38}$$

where

$$Q = \begin{bmatrix} BB^T + C_4 I & -BA^T & -BU^T \\ -AB^T & AA^T & AU^T \\ -UB^T & UA^T & UU^T \end{bmatrix} + \begin{bmatrix} E & -E & -E \\ -E & E & E \\ -E & E & E \end{bmatrix},$$

$$Q = Q^T, \quad S = \begin{bmatrix}\lambda\\ \alpha\\ \beta\end{bmatrix}, \quad r = \begin{bmatrix}0\\ e_1\\ (1-\varepsilon)e_r\end{bmatrix},$$

and E is a matrix with all entries equal to 1.

Algorithm 1 Linear IRUTSVM.

Input: $A \in R^{m_1 \times n}$, $B \in R^{m_2 \times n}$, $U \in R^{r \times n}$, where $r = m_2 - m_1 > 0$ and $g = \lceil \frac{m_1}{2} \rceil$.

1: Construct $B^* \in R^{m_1 \times n}$ and $U^* \in R^{g \times n}$ using randomly chosen reduced data samples of B and U, respectively.
2: Select the best parameters by using the grid search method. Determine hyperplanes (w_1, b_1), and (w_2, b_2) by solving the dual problems (36) and (38), respectively.
3: For classifying testing point x_i, if $\arg\min_{i=1,2} \frac{|x^T w_i + b_i|}{\|w_i\|} = 1$, then the data point belongs to class $+1$, otherwise the data point belongs to class -1.

By finding the optimal solution of the above dual problem and substituting in the following equations, the second hyperplane is derived

$$w_2 = -\frac{1}{C_4}\left(B^T\lambda - A^T\alpha - U^T\beta\right), \tag{39}$$

$$b_2 = -\frac{1}{C_4}\left(e_2^T\lambda - e_1^T\alpha - e_r^T\beta\right). \tag{40}$$

It is worth pointing out that our proposed method obviates the need for computing the inverse matrix, in contrast to other classical methods such as UTSVM, IUTSVM, and also RUTSVM.

A new data point x is assigned to class $i \in \{1, -1\}$ by the following decision rule:

$$\text{class } i = \arg\min_{i=1,2} \frac{|x^T w_i + b_i|}{\|w_i\|}.$$

The linear IRUTSVM is summarized in Algorithm 1.

3.2. Nonlinear IRUTSVM

In this subsection, we introduce a nonlinear version of the IRUTSVM, which is different to other usual methods. Indeed, for the nonlinear case of the IRUTSVM, we are seeking for two non-parallel hyperplanes to obtain the following decision functions:

$$f_1(x) = K(x^T, D^T)w_1 + b_1 \quad \text{and} \quad f_2(x) = K(x^T, D^T)w_2 + b_2, \tag{41}$$

where $D^T = [A^T, B^{*T}]$ and K is an appropriately chosen kernel function.

We used a nonlinear mapping $\Psi(\cdot)$ for the mapping of x into a higher dimensional feature space, i.e., $\Psi(\cdot): R^m \to R^l$, such that

$$X = \Psi(x),$$

where $X \in R^l$ and $l > m$. We call function K a kernel if there is a map $\Psi(\cdot)$ such that $K(x, y) = \langle \Psi(x), \Psi(y) \rangle$. So the first primal problem in nonlinear case can be expressed as follows

$$\begin{aligned}\min_{w_1,b_1,\xi,\psi,t}\ & \frac{1}{2}\|t\|^2 + \frac{c_2}{2}(\|w_1\|^2 + b_1^2) + c_1 e_1^T \xi + c_u e_g^T \psi \\ \text{s.t. } & \Psi(A)w_1 + e_1 b_1 = t, \\ & -(\Psi(B^*)w_1 + e_1 b_1) + \xi \ge e_1, \\ & (\Psi(U^*)w_1 + e_g b_1) + \psi \ge (-1+\varepsilon)e_g, \\ & \xi, \psi \ge 0, \end{aligned} \tag{42}$$

where c_1, $c_2 > 0$, $c_u > 0$ are penalty parameters, e_1, e_2, e_u are vectors of ones of appropriate dimensions, and ξ and ψ are slack vectors. Analogously, the second primal problem reads

$$\begin{aligned}\min_{w_2,b_2,\xi,\psi,t}\ & \frac{1}{2}\|t\|^2 + \frac{c_4}{2}(\|w_2\|^2 + b_2^2) + c_3 e_1^T \xi + c_u e_r^T \psi \\ \text{s.t. } & \Psi(B)w_2 + e_2 b_2 = t, \end{aligned}$$

$$\begin{aligned}&(\Psi(A)w_2 + e_1 b_2) + \xi \geq e_1,\\ &(\Psi(U)w_2 + e_r b_2) + \psi \geq (1-\varepsilon)e_r,\\ &\xi, \psi \geq 0,\end{aligned} \tag{43}$$

where c_3, $c_4 > 0$, $c_u > 0$ are parameters, e_1, e_2, e_r are vectors of ones of appropriate dimensions, and ξ and ψ are slack vectors.

Similarly to the linear case of IRUTSVM, the Wolfe dual problem of the problem (42) can be described as:

$$\begin{aligned}&\max_{\lambda,\alpha,\beta} \frac{1}{2}S^T QS + c_2 r^T S\\ &\text{s.t. } 0 \leq \alpha \leq c_1 e_1, \quad 0 \leq \beta \leq c_u e_g,\end{aligned} \tag{44}$$

where

$$Q = \begin{bmatrix} K(A,A^T)+C_2 I & K(A,B^{*T}) & -K(A,U^{*T}) \\ K(B^*,A^T) & K(B^*,B^{*T}) & -K(B^*,U^{*T}) \\ -K(U^*,A^T) & -K(U^*,B^{*T}) & K(U^*,U^{*T}) \end{bmatrix} + \begin{bmatrix} E & E & -E \\ E & E & -E \\ -E & -E & E \end{bmatrix},$$

$$Q = Q^T, \quad S = \begin{bmatrix}\lambda \\ \alpha \\ \beta\end{bmatrix}, \quad r = \begin{bmatrix} 0 \\ -e_1 \\ (1-\varepsilon)e_g \end{bmatrix},$$

and E is the matrix with all entries equal to 1. Notice that Q is a symmetric positive definite matrix. By solving the above dual problem, the first of nonlinear hyperplane is expressed as follows:

$$-\frac{1}{c_2}\left(K(x^T,A^T)\lambda + K(x^T,B^{*T})\alpha - K(x^T,U^{*T})\beta\right) + b_1 = 0, \tag{45}$$

where

$$b_1 = -\frac{1}{c_2}\left(e_1^T\lambda + e_1^T\alpha - e_g^T\beta\right).$$

Analogously, the Wolfe dual problem of the second primal problem (43) is:

$$\begin{aligned}&\max_{\lambda,\alpha,\beta} \frac{1}{2}S^T QS + c_4 r^T S\\ &\text{s.t. } 0 \leq \alpha \leq c_3 e_1, \quad 0 \leq \beta \leq c_u e_r,\end{aligned} \tag{46}$$

where

$$Q = \begin{bmatrix} K(B,B^T)+c_4 I & -K(B,A^T) & -K(B,U^T) \\ -K(A,B^T) & K(A,A^T) & K(A,U^T) \\ -K(U,B^T) & K(U,A^T) & K(U,U^T) \end{bmatrix} + \begin{bmatrix} E & -E & -E \\ -E & E & E \\ -E & E & E \end{bmatrix},$$

$$Q = Q^T, \quad S = \begin{bmatrix}\lambda \\ \alpha \\ \beta\end{bmatrix}, \quad r = \begin{bmatrix} 0 \\ -e_1 \\ (\varepsilon-1)e_r \end{bmatrix}.$$

By solving the above problem, the resulting nonlinear hyperplane takes the form of

$$\frac{1}{c_4}\left(K(x^T,B^T)\lambda - K(x^T,A^T)\alpha - K(x^T,U^T)\beta\right) + b_2 = 0, \tag{47}$$

where

$$b_2 = -\frac{1}{c_4}\left(e_2^T\lambda - e_1^T\alpha - e_r^T\beta\right).$$

Therefore we constructed two decision functions as follows:

$$f_1(x) = -\frac{1}{c_2}\left(K(x^T,A^T)\lambda + K(x^T,B^{*T})\alpha - K(x^T,U^{*T})\beta\right) + b_1, \tag{48}$$

Algorithm 2 NonLinear IRUTSVM.

Input: $A \in R^{m_1\times n}$, $B \in R^{m_2\times n}$, $U \in R^{r\times n}$, where $r = m_2 - m_1 > 0$, and $g = \lceil \frac{m_1}{2} \rceil$.

1: Construct $B^* \in R^{m_1\times n}$ and $U^* \in R^{g\times n}$ using randomly chosen reduced data samples of B and U, respectively.
2: Choose an appropriate kernel function K.
3: Select the best parameters by using the grid search method.
4: Determine decision functions $f_1(x)$, and $f_2(x)$ by (48) and (49), respectively, by solving their related dual problems.
5: For classifying testing point x_i, if $\arg\min_{i=1,2} |f_i(x)| = 1$, then the data point belongs to class $+1$, otherwise the data point belongs to class -1.

$$b_1 = -\frac{1}{c_2}\left(e_1^T\lambda + e_1^T\alpha - e_g^T\beta\right),$$

and

$$f_2(x) = -\frac{1}{c_4}\left(K(x^T,B^T)\lambda - K(x^T,A^T)\alpha - K(x^T,U^T)\beta\right) + b_2, \tag{49}$$

$$b_2 = -\frac{1}{c_4}\left(e_2^T\lambda - e_1^T\alpha - e_r^T\beta\right).$$

We have to note that for the nonlinear case, similarly to the linear case, our proposed method need not compute inverse matrices.

For any input value x, we assign it to the class $i \in \{+1, -1\}$ by the rule

$$\text{class } i = \arg\min_{i=1,2} |f_i(x)|.$$

The nonlinear IRUTSVM is summarized in Algorithm 2.

3.3. Computational complexity

Let m_1, m_2 and m_3 be the number of data samples in the positive, negative and universum classes, respectively. Assume that n is fixed, so we measure the complexity with respect to the number of samples, which is a common approach. The theoretical time complexity of TSVM is $O(m_1^3 + m_2^3)$ (Jayadeva et al., 2007). In UTSVM (Qi et al., 2012), universum samples increase the time complexity, hence, the overall time complexity of UTSVM is $O(m_1^3+m_2^3+m_3^3)$. The time complexity of RUTSVM is $O(m_1^3+m_2^3+m_3^3)$.

The proposed IRUTSVM needs to solve a convex quadratic programming problem (36) in dimension $m_1 + m_2 + m_3$; the other procedures are negligible. Therefore, the time complexity is $O((m_1+m_2+m_3)^3) = O(m_1^3+m_2^3+m_3^3)$. Using the imbalance ratio $IR = \frac{m_2}{m_1}$ and assuming that $IR > 1$, the time complexity can be equivalently expressed as $O(m_1^3+IR^3m_1^3+m_3^3) = O(IR^3m_1^3+m_3^3)$.

This means that, asymptotically, the time complexity of IRUTSVM is of the same order as UTSVM. Asymptotic time complexity, however, gives only an approximate figure about the real-life computational cost. Therefore, the practical time complexity is analyzed in the following section based on thorough numerical testing.

4. Numerical experiments

In this section, we analyze the experimental results of TSVM, UTSVM, RUTSVM and the proposed IRUTSVM on UCI datasets (Dua & Graff, 2019) and NDC datasets (Musicant, 1998). For the comparison of the models, we evaluate the area under the curve (AUC) and training time. To show the efficacy of the proposed IRUTSVM model, we evaluated the models on NDC datasets. To analyze the effect of hyperparameters, we evaluated the proposed IRUTSVM model at different values of C_1 and C_2.

All the experiments were conducted by a computer with these specifications: Windows-10 with 128-GB RAM Intel(R) Xeon(R)

Table 1
UCI dataset details.

Dataset	Train	Test	IR-All	IR-Trains
abalone9–18	(510 × 7)	(221 × 7)	16.4048	16.5862
brwisconsin	(477 × 9)	(206 × 9)	1.8577	1.8225
ecoli-0-1-4-6_vs_5	(195 × 6)	(85 × 6)	13	12
ecoli-0-1-4-7_vs_2-3-5-6	(234 × 7)	(102 × 7)	10.5862	14.6
ecoli-0-1-4-7_vs_5–6	(231 × 6)	(101 × 6)	12.28	13.4375
ecoli-0-1_vs_2-3-5	(169 × 7)	(75 × 7)	9.1667	7.45
ecoli-0-1_vs_5	(167 × 6)	(73 × 6)	11	8.2778
ecoli-0-2-3-4_vs_5	(140 × 7)	(62 × 7)	9.1	9
ecoli-0-2-6-7_vs_3–5	(155 × 7)	(69 × 7)	9.1818	10.9231
ecoli-0-3-4-6_vs_5	(142 × 7)	(63 × 7)	9.25	9.9231
ecoli-0-3-4-7_vs_5–6	(178 × 7)	(79 × 7)	9.28	7.4762
ecoli-0-4-6_vs_5	(141 × 6)	(62 × 6)	9.15	9.0714
ecoli-0-6-7_vs_3–5	(154 × 7)	(68 × 7)	9.0909	13
ecoli-0-6-7_vs_5	(153 × 6)	(67 × 6)	10	16
ecoli0137vs26	(216 × 7)	(95 × 7)	4.7593	4.6842
ecoli01vs5	(167 × 7)	(73 × 7)	11	12.9167
ecoli4	(234 × 7)	(102 × 7)	15.8	15.7143
glass2	(148 × 9)	(66 × 9)	11.5882	11.3333
glass4	(148 × 9)	(66 × 9)	15.4615	13.8
glass5	(148 × 9)	(66 × 9)	22.7778	28.6
haber	(213 × 3)	(93 × 3)	2.7778	2.9444
haberman	(213 × 3)	(93 × 3)	2.7778	2.9444
iono	(244 × 33)	(107 × 33)	1.7857	1.8372
led7digit-0-2-4-5-6-7-8-9_vs_1	(309 × 7)	(134 × 7)	10.973	10.8846
monk2	(419 × 7)	(182 × 7)	1.9175	1.9301
new-thyroid1	(149 × 5)	(66 × 5)	5.1429	5.7727
segment0	(1614 × 19)	(694 × 19)	6.0152	5.839
shuttle-6_vs_2–3	(160 × 9)	(70 × 9)	22	19
shuttle-c0-vs-c4	(1279 × 9)	(550 × 9)	13.8699	12.7527
transfusion	(522 × 4)	(226 × 4)	3.2022	3.5
vehicle1	(591 × 18)	(255 × 18)	2.8986	2.8627
vowel	(690 × 10)	(298 × 10)	9.9778	9.4545
yeast-0-2-5-6_vs_3-7-8-9	(701 × 8)	(303 × 8)	9.1414	8.8732
yeast-0-5-6-7-9_vs_4	(368 × 8)	(160 × 8)	9.3529	7.9756
yeast-2_vs_4	(358 × 8)	(156 × 8)	9.0784	10.9333
yeast1	(2076 × 8)	(892 × 8)	2.4592	2.4201
yeast1vs7	(320 × 8)	(139 × 8)	14.3	17.8235
yeast2vs8	(337 × 8)	(146 × 8)	23.15	20.0625
yeast5	(1037 × 8)	(447 × 8)	32.7273	31.4063

CPU E5-2697 v4 2.30 GHz with MATLAB R2017b. The Gaussian kernel, $K(x, y) = \exp(-(\|x - y\|^2/\mu^2))$ was also used, where μ is a hyperparameter. The data was divided into 70 : 30 ratio for testing and training the models, respectively. A 5-fold cross-validation was applied to the training data in order to obtain the optimal parameters of different models via grid search approach. In 5-fold cross-validation, the datasets are separated into five equally-sized disjoint subsets, and the training of classifier is performed on all the subsets except with the exception of one, which is called test data. The dual problems which are quadratic programming problems are solved by using `quadprog.m` in MATLAB.

We evaluate the performance of the classification models via the area under the curve and the training times. Mathematically, Accuracy or AUC is defined as

$$\text{AUC} = \frac{TP + TN}{TP + FP + TN + FN}, \tag{50}$$

where TN is the true negative rate, TP is the true positive rate, FN is the false negative rate, and FP is the false positive rate.

4.1. Choosing parameters

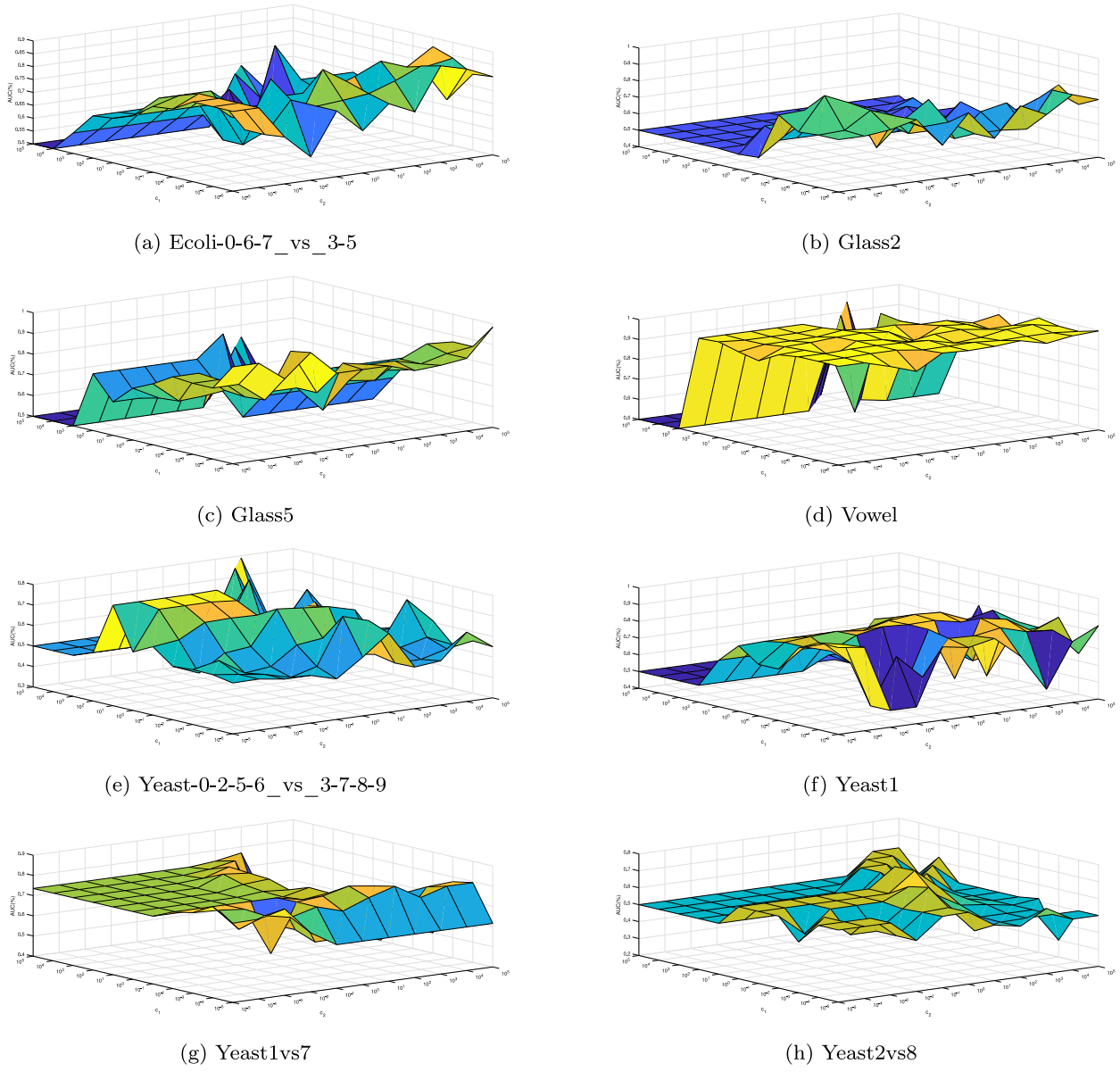
To analyze the effect of parameters C_1 and C_2 of the proposed IRUTSVM models, we plot the accuracy measure across the given range of parameters. Fig. 1 shows the effect of the hyperparameters. Hence, to get a better generalization performance, the hyperparameters need to be chosen carefully. Therefore, the grid search method was adopted to choose the best values of the parameters for the given algorithms (Hsu, Chang, & Lin, 2003).

The optimal parameters were obtained via grid search approach from the following range: $c_0 = [0.5, 1, 1.5, 2, 2.5]$, $\mu = [10^{-5}, 10^{-4}, \ldots, 10^4, 10^5]$, $C = C_1 = C_2 = C_3 = C_4 = C_u = [10^{-5}, 10^{-4}, \ldots, 10^4, 10^5]$.

4.2. Experimental analysis

Table 1 summarizes the details of the binary datasets used for the evaluation of the classification models. The first column displays the name, the second column depicts the training data size, and the third column shows the testing data size. The last two columns record the imbalance ratio present in the whole data and the training data, respectively.

Table 2 summarizes the classification efficiency of the presented IRUTSVM model as well as the baseline models. In ecoli-0-1-4-7_vs_2-3-5-6, the proposed IRUTSVM model achieved 91.72% accuracy followed by TSVM with 90% while as UTSVM and RUTSVM are less than 90% accuracy. In ecoli-0-1-4-7_vs_5-6, the proposed IRUTSVM model achieved 93.36% accuracy followed by RUTSVM with 92.81% accuracy while as other model achieved less than 90% accuracy. Similarly, in glass2, led7digit-0-2-4-5-6-7-8-9_vs_1, vehicle1 and yeast1vs7 the proposed models show better accuracy compared to the baseline models. As is clear, the proposed IRUTSVM model outperforms the baseline models. The average accuracy of the proposed IRUTSVM is 0.8701 which is at least 3% better, compared to the baseline models. Since the average accuracy measure may not be good for the evaluation of the models as the poor performance in some datasets may be compensated in other models. Hence, for an accurate comparison, the ranking of the models was used on each dataset.



(a) Ecoli-0-6-7_vs_3-5 (b) Glass2

(c) Glass5 (d) Vowel

(e) Yeast-0-2-5-6_vs_3-7-8-9 (f) Yeast1

(g) Yeast1vs7 (h) Yeast2vs8

Fig. 1. The performance of the proposed IRUTSVM model with the varying hyperparameters C_1 and C_2.

Here, the performance of each model is ranked on each dataset with the lower performing model getting the higher rank and the higher-performing models being assigned the lower rank. Hence, the lower rank of the model indicates the better generalization performance of the model. According to Table 2, the mean rank of the presented IRUTSVM model is 2.0769, followed by UTSVM with 2.4744, TSVM with 2.6154 and RUTSVM with 2.8333. As the average rank is lowest, the presented IRUTSM model outperforms the given models in terms of generalization performance.

We used David Musicant's NDC Data Generator (Musicant, 1998) to generate NDC datasets, which have normal distribution. Moosaei, Musicant, Khosravi, and Hladík (2020) presented an extended version of these datasets with arbitrary numbers of samples, features, and classes. For a deeper analysis of the algorithms and evaluating models' performance on large-scale datasets, we evaluated the models on NDC datasets. Table 3 summarizes presented IRUTSVM model's performance and the baseline models. According to Table 3, the IRUTSVM model is more accurate than the baseline models. With respect to the training time, the proposed IRUTSVM models takes more time for training the model compared to the baseline models. The possible reason is that the final optimization problem in the proposed IRUTSVM model involves the variables of sample size (which involves the positive, negative and universum samples).

4.3. Statistical analysis

To evaluate the performance of the models statistically, the Friedman test (Demšar, 2006) was used. Here, we evaluated the performance of 4 classification models on 39 datasets.

Let m classification models be ranked on d datasets, then the Friedman statistic is given by

$$\chi_F^2 = \frac{12d}{m(m+1)}\left[\sum_j S_j^2 - \frac{m(m+1)^2}{4}\right]; \tag{51}$$

it is distributed with χ_F^2 with $(m-1)$ degrees of freedom. Since χ_F^2 is undesirably conservative, another suitable statistic is given

Table 2
Classification accuracy of TSVM, UTSVM, RUTSVM and the proposed IRUTSVM with Gaussian kernel.

Dataset	TSVM (AUC, Time (s)) (C, μ)	UTSVM (AUC, Time (s)) (C, ϵ, μ)	RUTSVM (AUC, Time (s)) (C, ϵ, μ)	IRUTSVM (AUC, Time (s)) (C, ϵ, μ)
abalone9–18	(**0.8295**, 0.12) (100, 1)	(0.7692, 0.1386) (0.1, 0.6, 10^{-3})	(0.5649, 0.0739) (100, 0.1, 10)	(0.75, 0.135) (10^{-4}, 0.1, 10^{2})
brwisconsin	(0.9685, 0.0746) (10^{-3}, 10^{2})	(0.9637, 0.1355) (10^{-5}, 0.1, 10^{2})	(**0.9745**, 0.0638) (10, 0.5, 10^{5})	(0.9743, 1.9109) (10, 0.3, 10^{-2})
ecoli-0-1-4-6_vs_5	(0.9877, 0.0391) (10, 10^{5})	(**0.9938**, 0.0439) (10^{-2}, 0.5, 10^{3})	(0.9875, 0.0151) (10^{-3}, 0.6, 10^{3})	(0.9875, 0.0281) (10^{-2}, 0.1, 10^{-3})
ecoli-0-1-4-7_vs_2-3-5-6	(0.9, 0.0387) (1, 10^{4})	(0.8929, 0.0466) (1, 0.1, 10^{4})	(0.8287, 0.0251) (1, 0.6, 10^{4})	(**0.9172**, 0.0201) (10^{-5}, 0.1, 10^{-3})
ecoli-0-1-4-7_vs_5–6	(0.875, 0.044) (10, 10^{5})	(0.8889, 0.0331) (1, 0.1, 10^{4})	(0.9281, 0.0181) (1, 0.1, 10^{4})	(**0.9336**, 0.3946) (0.1, 0.3, 10^{-4})
ecoli-0-1_vs_2-3-5	(0.7143, 0.0389) (0.1, 10^{4})	(**0.875**, 0.0379) (1, 0.5, 10^{4})	(0.8539, 0.0141) (0.1, 0.6, 10^{5})	(0.868, 0.0256) (10^{-4}, 0.1, 10^{-4})
ecoli-0-1_vs_5	(0.8333, 0.037) (1, 10^{5})	(**0.9577**, 0.0281) (100, 0.3, 10^{4})	(**0.9577**, 0.0154) (1, 0.3, 10^{4})	(**0.9577**, 0.0395) (10^{-3}, 0.3, 10^{-4})
ecoli-0-2-3-4_vs_5	(**0.9737**, 0.0368) (10, 10^{4})	(0.8244, 0.0134) (10, 0.1, 10^{4})	(0.8244, 0.0131) (10^{-2}, 0.3, 10^{3})	(0.8244, 0.0374) (10^{-3}, 0.1, 10^{-3})
ecoli-0-2-6-7_vs_3–5	(**1**, 0.0447) (1, 10^{4})	(0.8722, 0.0239) (10, 0.5, 10^{4})	(0.8389, 0.0224) (1, 0.6, 10^{4})	(0.9278, 0.0202) (10^{-4}, 0.1, 10^{-4})
ecoli-0-3-4-6_vs_5	(0.8333, 0.0411) (10^{-2}, 10^{4})	(0.9107, 0.02) (1, 0.1, 10^{4})	(0.9018, 0.0141) (10^{-2}, 0.6, 10^{4})	(**0.9196**, 0.0352) (10^{-2}, 0.3, 10^{-3})
ecoli-0-3-4-7_vs_5–6	(0.8428, 0.0414) (1, 10^{3})	(**1**, 0.0261) (0.1, 0.1, 10^{4})	(**1**, 0.0202) (1, 0.5, 10^{5})	(0.98, 0.062) (10^{-2}, 0.5, 10^{-4})
ecoli-0-4-6_vs_5	(0.8571, 0.0432) (0.1, 10^{4})	(**1**, 0.0139) (10, 0.1, 10^{4})	(0.9167, 0.0137) (1, 0.1, 10^{3})	(**1**, 0.0185) (10^{-4}, 0.3, 10^{-4})
ecoli-0-6-7_vs_3–5	(**0.9375**, 0.0411) (1, 10^{4})	(0.8006, 0.0252) (10, 0.3, 10^{4})	(0.7097, 0.0146) (10^{-2}, 0.6, 10^{3})	(0.8389, 0.0833) (100, 0.5, 10^{-3})
ecoli-0-6-7_vs_5	(0.7418, 0.0389) (10^{-2}, 10^{3})	(0.7273, 0.0256) (10^{-2}, 0.1, 10^{4})	(**0.8636**, 0.0191) (0.1, 0.1, 10^{5})	(0.7273, 0.0124) (10^{-5}, 0.1, 10^{-4})
ecoli0137vs26	(**0.9412**, 0.0416) (1, 1)	(0.8374, 0.032) (10^{-2}, 0.1, 1)	(0.8497, 0.0166) (0.1, 0.6, 0.1)	(0.8687, 0.2149) (0.1, 0.3, 1)
ecoli01vs5	(**1**, 0.0447) (10^{-2}, 0.1)	(0.875, 0.0247) (10^{-3}, 0.6, 10^{-2})	(0.9298, 0.0204) (10^{-5}, 0.1, 10)	(0.9923, 0.0135) (10^{-5}, 0.1, 1)
ecoli4	(**1**, 0.0434) (10, 10^{2})	(0.9896, 0.0349) (10, 0.5, 10^{2})	(0.9115, 0.0139) (10^{-2}, 0.1, 10)	(0.9896, 0.155) (0.1, 0.3, 0.1)
glass2	(0.5927, 0.0398) (100, 10^{5})	(0.6639, 0.0212) (10^{-2}, 0.1, 10^{3})	(0.5033, 0.0141) (10, 0.5, 10^{2})	(**0.8426**, 0.0278) (10^{-3}, 0.5, 1)
glass4	(**0.8918**, 0.04) (1, 10^{2})	(0.6508, 0.0242) (10^{-3}, 0.6, 1)	(0.8016, 0.0206) (10, 0.1, 10^{3})	(0.7381, 0.0154) (10^{-3}, 0.3, 10^{-5})
glass5	(0.75, 0.0389) (1, 10)	(**0.9677**, 0.0237) (10, 0.6, 10^{3})	(0.75, 0.0179) (1, 0.6, 10^{2})	(**0.9677**, 0.0128) (10^{-5}, 0.1, 0.1)
haber	(0.5106, 0.0407) (10, 10^{3})	(0.5833, 0.0325) (0.1, 0.6, 10)	(**0.6793**, 0.0186) (1, 0.6, 10^{3})	(0.5968, 0.0246) (10^{-5}, 0.1, 0.1)
haberman	(0.5106, 0.0466) (10, 10^{3})	(0.5833, 0.0321) (0.1, 0.6, 10)	(**0.6793**, 0.0214) (1, 0.6, 10^{3})	(0.5968, 0.0237) (10^{-5}, 0.1, 0.1)
iono	(**0.9575**, 0.0475) (10^{-3}, 1)	(0.9151, 0.0387) (10^{-4}, 0.1, 1)	(0.8278, 0.0226) (10^{-2}, 0.3, 10^{2})	(0.9302, 0.153) (1, 0.5, 1)
led7digit-0-2-4-5-6-7-8-9_vs_1	(0.8895, 0.0678) (100, 10^{3})	(0.9309, 0.0453) (0.1, 0.3, 10^{3})	(0.9146, 0.0265) (1, 0.1, 10^{3})	(**0.9383**, 0.4732) (1, 0.1, 10^{-2})
monk2	(0.7349, 0.0918) (10, 10^{2})	(**0.7736**, 0.1164) (10, 0.1, 10^{2})	(0.7045, 0.0659) (1, 0.3, 10^{2})	(0.712, 0.2014) (10^{-5}, 0.6, 10^{-2})

(*continued on next page*)

as follows:

$$F_F = \frac{(d-1)\chi_F^2}{d(m-1)-\chi_F^2}. \tag{52}$$

F_F follows $F-$distribution with $(m-1)$ and $(m-1)(d-1)$ degrees of freedom. Under the Null hypothesis, all the models perform equally, hence their average ranks are equal.

The average rank of the TSVM, UTSVM, RUTSVM and the proposed IRUTSVM model are 2.6154, 2.4744, 2.8333 and 2.0769, respectively. By a simple calculation, we obtain $\chi_F^2 = 7.1154$ and $F_F = 2.4606$, where F_F is distributed with $(m-1) = (4-1) = 3$ and $(m-1)(d-1) = (4-1)(39-1) = 114$ degrees of freedom. At 5% level of significance, $F_F(3, 114) = 2.69$. Hence, the Friedman test is unable to demonstrate significant differences between the presented IRUTSVM model and the baseline models. However, as is evident, the proposed IRUTSVM model outperforms baseline models generalization performance because the proposed IRUTSVM model achieved the highest average accuracy and lowest average rank.

We also considered pairwise win-tie-loss performance to examine models' performance in a pairwise manner. Table 4 summarizes the pairwise results. One can see that the proposed IRUTSVM model wins 22 times, losses 15 times, and ties 2-times concerning the TSVM model. Concerning the UTSVM model, the proposed IRUTSVM model wins 21 times, losses 10 times, and ties 8-times. The proposed model achieved the highest number of wins concerning the RUTSVM model where it showed 25 wins, followed by 10 losses and 4 ties.

5. Conclusions

Recently, a RUTSVM has been presented for learning class imbalance. The RUTSVM incorporates prior information from the

Table 2 (*continued*).

Dataset	TSVM (AUC, Time(s)) (C_1, μ)	UTSVM (AUC, Time(s)) (C_1, ϵ, μ)	RUTSVM (AUC, Time(s)) (C_1, ϵ, μ)	IRUTSVM (AUC, Time(s)) (C_1, ϵ, μ)
new-thyroid1	(**1**, 0.04) (1, 10^3)	(**1**, 0.0185) (10^{-4}, 0.1, 10^4)	(0.9811, 0.0173) (1, 0.3, 10^2)	(**1**, 0.1055) (0.1, 0.5, 10^{-2})
segment0	(0.9892, 1.1866) (10, 10^4)	(**0.9983**, 1.0041) (10^{-3}, 0.1, 10^3)	(0.9921, 0.4843) (10^{-2}, 0.3, 10^4)	(0.9942, 1.4287) (10^{-5}, 0.3, 10^{-3})
shuttle-6_vs_2–3	(0.75, 0.0381) (10^{-5}, 10^5)	(0.75, 0.0216) (10^{-5}, 0.1, 10^5)	(**1**, 0.0135) (10^{-4}, 0.1, 10^5)	(0.9926, 0.0119) (10^{-5}, 0.1, 10^{-5})
shuttle-c0-vs-c4	(**1**, 1.0359) (10^{-2}, 10^5)	(0.9833, 0.7141) (10^{-3}, 0.6, 10^4)	(**1**, 0.3532) (1, 0.6, 10^4)	(**1**, 2.3238) (10^{-3}, 0.1, 10^{-4})
transfusion	(0.6255, 0.1423) (10, 10^5)	(**0.6437**, 0.1614) (0.1, 0.5, 10^3)	(0.6117, 0.0704) (0.1, 0.6, 10^3)	(0.6404, 0.1905) (10^{-4}, 0.1, 10^{-2})
vehicle1	(0.8271, 0.1426) (100, 10^5)	(0.8037, 0.199) (100, 0.3, 10^5)	(0.8039, 0.1046) (10, 0.5, 10^4)	(**0.8744**, 4.2909) (0.1, 0.1, 10^{-4})
vowel	(0.7778, 0.2086) (10^{-4}, 1)	(0.9982, 0.2138) (10^{-4}, 0.1, 1)	(0.8904, 0.151) (1, 0.3, 10)	(**1**, 0.8378) (10^{-2}, 0.6, 0.1)
yeast-0-2-5-6_vs_3-7-8-9	(0.6954, 0.2627) (10, 10^2)	(0.7566, 0.1793) (10^{-2}, 0.1, 10^2)	(**0.7742**, 0.099) (0.1, 0.5, 10^2)	(0.7176, 4.2649) (0.1, 0.1, 1)
yeast-0-5-6-7-9_vs_4	(0.7186, 0.0871) (100, 10^5)	(0.8267, 0.0657) (10^{-4}, 0.1, 1)	(**0.8533**, 0.0376) (1, 0.6, 10^2)	(0.8233, 0.1183) (10^{-2}, 0.5, 10^{-3})
yeast-2_vs_4	(**0.836**, 0.0688) (100, 1)	(0.8148, 0.0664) (10, 0.1, 0.1)	(0.7815, 0.0288) (1, 0.3, 10^2)	(0.8254, 0.0537) (10^{-5}, 0.1, 0.1)
yeast1	(**0.9917**, 2.0328) (10, 10^{-2})	(0.8676, 2.2234) (10^{-4}, 0.5, 10^{-2})	(0.8661, 0.9254) (10^{-5}, 0.3, 10^{-2})	(0.9183, 3.3138) (10^{-5}, 0.5, 10^2)
yeast1vs7	(0.5846, 0.0729) (100, 10)	(0.7549, 0.0593) (10^{-2}, 0.5, 10)	(0.6517, 0.0242) (10^{-5}, 0.1, 10)	(**0.7802**, 0.0458) (10^{-3}, 0.6, 10^{-4})
yeast2vs8	(**0.7218**, 0.0716) (100, 0.1)	(0.625, 0.0622) (1, 0.1, 0.1)	(0.6144, 0.0306) (10^{-2}, 0.6, 1)	(0.625, 0.1729) (100, 0.1, 0.1)
yeast5	(0.9198, 0.7663) (1000, 10^4)	(**0.9828**, 0.4328) (10^{-2}, 0.1, 10^2)	(0.9667, 0.1891) (10^{-5}, 0.1, 10^2)	(0.9644, 1.4536) (100, 0.5, 10^{-2})
Average Accuracy	0.8336	0.8475	0.833	**0.8701**
Average Rank	2.6154	2.4744	2.8333	**2.0769**
Overall Win-Tie-Loss	12-0-15	6-0-6	7-0-13	8-0-0

Table 3
Performance on NDC datasets with Gaussian Kernel.

Dataset	TSVM (AUC, Time(s))	UTSVM (AUC, Time(s))	RUTSVM (AUC, Time(s))	IRUTSVM (AUC, Time(s))
NDC-6k	(0.8043, 18.2419)	(0.9078, 30.2271)	(0.9295, 13.2867)	(**0.9647**, 54.9582)
NDC-10k	(0.9595, 69.6511)	(0.9595, 113.29)	(0.965, 45.2359)	(**1**, 166.179)
NDC-15k	(0.8825, 210.698)	(0.9524, 348.741)	(0.9524, 118.959)	(**0.9891**, 363.254)
NDC-35k	(0.9724, 2228.62)	(0.9833, 4974.5)	(0.9809, 1590.89)	(**0.9907**, 7494.45)
NDC-40k	(0.9272, 3624.45)	(0.9715, 13021.9)	(0.9715, 1567.54)	(**0.9946**, 20093.2)

Table 4
Pairwise win tie loss analysis. Here, $a-b-c$ means that the row method wins a-times, ties b-times and loses c-times with respect to the column method of the corresponding cell.

	TSVM	UTSVM	RUTSVM
UTSVM	20-2-17		
RUTSVM	18-2-19	13-3-23	
IRUTSVM	22-2-15	21-8-10	25-4-10

universum data and creates a balance situation for the classification problem. Although RUTSVM can classify imbalanced datasets, it has two drawbacks that need to be addressed. The First, which is expensive and error-prone, is inverse matrix calculation. Also, the RUTSVM introduced a small positive value for obtaining classifiers to prevent singular matrices, which implies classifiers are not very precise. Second, if we apply the linear kernel to the nonlinear RUTSVM, we will not have to deal with a problem that is comparable to the linear scenario. This paper proposed an optimization technique for solving RUTSVM and overcoming its related issues. In this paper, we improved the RUTSVM by introducing the different but equivalent form of the primal problems so that the related Lagrangian functions are slightly different and we call the method as IRUTSVM. In contrast to RUTSVM, the presented method does not compute large inverse matrices that we have in a wide range of classical algorithms. We also suggest a kernel trick for the nonlinear IRUTSVM, which is different from the existing RUTSVM, so that theoretically and numerically the nonlinear IRUTSVM outperforms the nonlinear RUTSVM. The numerical experiments performed on several benchmark imbalanced datasets illustrate that the presented IRUTSVM method has desirable generalization performance. Therefore, our proposed method not only overcomes the two drawbacks but also increases the classification accuracy.

CRediT authorship contribution statement

Hossein Moosaei: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **M.A. Ganaie:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft. **Milan Hladík:** Conceptualization, Formal analysis, Funding acquisition, Project administration, Resources, Supervision, Validation, Visualization, Writing – review & editing. **M. Tanveer:** Conceptualization,

Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – review & editing.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Acknowledgments

The work of H. Moosaei was supported by Center for Foundations of Modern Computer Science (Charles Univ. project UNCE/SCI/004). The work of M. Hladík was supported by the Czech Science Foundation Grant P403-22-11117S. This work is also supported by Science and Engineering Research Board (SERB), India under Mathematical Research Impact-Centric Support (MATRICS) scheme Grant No. MTR/2021/000787.

References

Arabasadi, Z., Alizadehsani, R., Roshanzamir, M., Moosaei, H., & Yarifard, A. A. (2017). Computer aided decision making for heart disease detection using hybrid neural network-Genetic algorithm. *Computer Methods and Programs in Biomedicine*, *141*, 19–26.

Bazikar, F., Ketabchi, S., & Moosaei, H. (2020). DC programming and DCA for parametric-margin ν-support vector machine. *Applied Intelligence*, *50*(6), 1763–1774.

Cai, Y. D., Ricardo, P. W., Jen, C. H., & Chou, K. C. (2004). Application of SVM to predict membrane protein types. *Journal of Theoretical Biology*, *226*(4), 373–376.

Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, *408*, 189–215.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, *6*(1), 1–6.

Chen, W. J., Shao, Y. H., Li, C. N., Liu, M. Z., Wang, Z., & Deng, N. Y. (2020). ν-Projection twin support vector machine for pattern classification. *Neurocomputing*, *376*, 10–24.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Dua, D., & Graff, C. (2019). UCI machine learning repository, 2017. *37*, URL http://archiveics.uci.edu/ml.

Ganaie, M. A., & Tanveer, M. (2020). LSTSVM classifier with enhanced features from pre-trained functional link network. *Applied Soft Computing*, *93*, Article 106305.

Ganaie, M. A., Tanveer, M., & Lin, C. T. (2022). Large scale fuzzy least squares twin SVMs for class imbalance learning. *IEEE Transactions on Fuzzy Systems*.

Ganaie, M. A., Tanveer, M., & Suganthan, P. N. (2020). Oblique decision tree ensemble via twin bounded SVM. *Expert Systems with Applications*, *143*, Article 113072.

Gupta, D., Sarma, H. J., Mishra, K., & Prasad, M. (2019). Regularized universum twin support vector machine for classification of EEG signal. In *2019 IEEE International Conference on Systems, Man and Cybernetics* (pp. 2298–2304). IEEE.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A practical guide to support vector classification. URL https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Huang, H., Wei, X., & Zhou, Y. (2018). Twin support vector machines: A survey. *Neurocomputing*, *300*, 34–43.

Javadi, S. H., Moosaei, H., & Ciuonzo, D. (2019). Learning wireless sensor networks for source localization. *Sensors*, *19*(3), 635.

Jayadeva, Khemchandani, R., & Chandra, S. (2007). Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(5), 905–910.

Ketabchi, S., Moosaei, H., Razzaghi, M., & Pardalos, P. M. (2019). An improvement on parametric ν-support vector algorithm for classification. *Annals of Operations Research*, *276*(1–2), 155–168.

Kumar, M. A., & Gopal, M. (2009). Least squares twin support vector machines for pattern classification. *Expert Systems with Applications*, *36*(4), 7535–7543.

Kumar, B., & Gupta, D. (2021). Universum based Lagrangian twin bounded support vector machine to classify EEG signals. *Computer Methods and Programs in Biomedicine*, *208*, Article 106244.

Ma, J., Yang, L., & Sun, Q. (2020). Capped L1-norm distance metric-based fast robust twin bounded support vector machine. *Neurocomputing*, *412*, 295–311.

Mangasarian, O. L., & Wild, E. W. (2005). Multisurface proximal support vector machine classification via generalized eigenvalues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(1), 69–74.

Moosaei, H., Ketabchi, S., Razzaghi, M., & Tanveer, M. (2021). Generalized twin support vector machines. *Neural Processing Letters*, 1–20.

Moosaei, H., Musicant, D., Khosravi, S., & Hladík, M. (2020). MC-NDC: multi-class normally distributed clustered datasets. URL https://github.com/dmusican/ndc.

Musicant, D. R. (1998). NDC: normally distributed clustered datasets. URL https://research.cs.wisc.edu/dmi/svm/ndc/.

Noble, W. S. (2004). Support vector machine applications in computational biology. In B. Schoelkopf, K. Tsuda, & J.-P. Vert (Eds.), *Computational molecular biology*, *Kernel methods in computational biology* (pp. 71–92). MIT Press.

Qi, Z., Tian, Y., & Shi, Y. (2012). Twin support vector machine with universum data. *Neural Networks*, *36*, 112–119.

Richhariya, B., & Gupta, D. (2019). Facial expression recognition using iterative universum twin support vector machine. *Applied Soft Computing*, *76*, 53–67.

Richhariya, B., Sharma, A., & Tanveer, M. (2018). Improved universum twin support vector machine. In *2018 IEEE symposium series on computational intelligence* (pp. 2045–2052). IEEE.

Richhariya, B., & Tanveer, M. (2018). EEG signal classification using universum support vector machine. *Expert Systems with Applications*, *106*, 169–182.

Richhariya, B., & Tanveer, M. (2020). A reduced universum twin support vector machine for class imbalance learning. *Pattern Recognition*, *102*, Article 107150.

Richhariya, B., Tanveer, M., & Initiative, A. D. N. (2020). An efficient angle based universum least squares twin support vector machine for pattern classification. In *ACM transactions on internet technology*.

Richhariya, B., Tanveer, M., Rashid, A., & Initiative, A. D. N. (2020). Diagnosis of Alzheimer's disease using universum support vector machine based recursive feature elimination (USvm-RFE). *Biomedical Signal Processing and Control*, *59*, Article 101903.

Shao, Y. H., Chen, W. J., Zhang, J. J., Wang, Z., & Deng, N. Y. (2014). An efficient weighted Lagrangian twin support vector machine for imbalanced data classification. *Pattern Recognition*, *47*(9), 3158–3167.

Shao, Y. H., Zhang, C. H., Wang, X. B., & Deng, N. Y. (2011). Improvements on twin support vector machines. *IEEE Transactions on Neural Networks*, *22*(6), 962–968.

Tang, Y., Zhang, Y. Q., Chawla, N. V., & Krasser, S. (2008). SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, *39*(1), 281–288.

Tanveer, M., Gautam, C., & Suganthan, P. N. (2019). Comprehensive evaluation of twin SVM based classifiers on UCI datasets. *Applied Soft Computing*, *83*, Article 105617.

Tanveer, M., Khan, M. A., & Ho, S. S. (2016). Robust energy-based least squares twin support vector machines. *Applied Intelligence*, *45*(1), 174–186.

Tanveer, M., Rajani, T., Rastogi, R., Shao, Y. H., & Ganaie, M. A. (2022). Comprehensive review on twin support vector machines. *Annals of Operations Research*, 1–46.

Tanveer, M., Sharma, A., & Suganthan, P. N. (2019). General twin support vector machine with pinball loss function. *Information Sciences*, *494*, 311–327.

Tanveer, M., Tiwari, A., Choudhary, R., & Ganaie, M. A. (2021). Large-scale pinball twin support vector machines. *Machine Learning*, 1–24.

Tong, S., & Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, *2*(Nov), 45–66.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer.

Vapnik, V. N., & Chervonenkis, A. J. (1974). *Theory of pattern recognition*. Moscow: Nauka.

Wang, X. Z., He, Q., Chen, D. G., & Yeung, D. (2005). A genetic algorithm for solving the inverse problem of support vector machines. *Neurocomputing*, *68*, 225–238.

Wang, X. Y., Wang, T., & Bu, J. (2011). Color image segmentation using pixel wise support vector machine classification. *Pattern Recognition*, *44*(4), 777–787.

Wang, X. Z., Xing, H. J., Li, Y., Hua, Q., Dong, C. R., & Pedrycz, W. (2014). A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning. *IEEE Transactions on Fuzzy Systems*, *23*(5), 1638–1654.

Weston, J., Collobert, R., Sinz, F., Bottou, L., & Vapnik, V. (2006). Inference with the universum. In *Proceedings of the 23rd international conference on machine learning* (pp. 1009–1016).

Xu, Y., Chen, M., & Li, G. (2016). Least squares twin support vector machine with Universum data for classification. *International Journal of Systems Science*, *47*(15), 3637–3645.