Soft Computing (2022) 26:10049–10062
https://doi.org/10.1007/s00500-022-07376-2

DATA ANALYTICS AND MACHINE LEARNING

Support vector machine classification using semi-parametric model

Mohammad Ghassem Akbari[1] · Saeed Khorashadizadeh[2] · Mohammad-Hassan Majidi[2]

Accepted: 5 July 2022 / Published online: 22 August 2022

Abstract
Pattern recognition and data mining using support vector machine (SVM) have been the focus of widespread researches in recent decades. In SVM, a hyper-plane is designed to classify the training data. A challenge in SVM is that the parameters of hyper-planes are constants. As a result, there may be some critical points that will be classified into a wrong set. It should be mentioned that finding this hyper-plane is very similar to solving a regression problem using parametric or semi-parametric models in statistics. This is the main motivation of this paper. The contribution of this paper is combining SVM classifier and semi-parametric models (SP-SVM) to solve the aforementioned challenge. In fact, using semi-parametric linear model results in some serial linear decision boundaries with several slopes and intercepts. In other words, there are two types of kernels in the proposed SP-SVM: the kernels that perform nonlinear transformation of the input features and the kernels needed in the semi-parametric model. The validations have been done on Iris data set and also some other linearly non-separable classification problems. The accuracy of the proposed SP-SVM outperforms some related algorithms such as K-nearest neighbor (KNN)-based weighted multi-class twin support vector machines (KWMTSVM), support vector classification–regression machine for *K*-class classification (*K*-SVCR), twin multi-class classification support vector machines (twin-KSVC), intelligent particle swarm classifier (IPS-classifier) and random forest. The accuracy of SP-SVM is 97.33%. Thus, SP-SVM can play an important role in increasing the accuracy of industrial machines that perform classifications, for example, agricultural products.

Keywords Support vector machine · Semi-parametric linear regression · Decision boundary

1 Introduction

The support vector machine (SVM) is a learning method, which avoids the problems of over-learning, under-learning and local minimum of network structure in the artificial neural networks. As a binary classification algorithm, it has widespread applications in various fields of engineering and medical sciences (Subasi 2013; Dukart et al. 2013; Alajlan et al. 2012; Haddi et al. 2013; Roy et al. 2016). Time series prediction using SVM has been one of popular research areas in recent years. The SVMs proposed for time series prediction cover many practical application areas from financial market prediction to electric consumption load forecasting and other scientific fields (Sapankevych and Sankar 2009). Pattern recognition is also among the research fields that SVM has been applied frequently (Byun and Lee 2002; Bashbaghi et al. 2017; Christlein et al. 2017; Liu et al. 2018; Shah et al. 2017; Solera-Urena et al. 2012). For example, automatic speech recognition using SVM has been presented in Solera-Urena et al. (2012). In communication systems, various problems can be solved by SVM. For example, in wireless sensor networks, SVM can be used for solving the localization problems (Zhu and Wei 2017). In (Garcia et al. 2006), a robust SVM has been proposed for channel estimation in orthogonal frequency division multiplexing (OFDM) systems.

From mathematical point of view, SVM is based on the structural risk minimized using the maximum margin idea (Ozer et al. 2011). In fact, a convex objective function is optimized to find the classifier (decision boundary). Kernel function plays a dominant role in SVM generalization performance. To be more precise, kernel function transfers

✉ Saeed Khorashadizadeh
s.khorashadizadeh@birjand.ac.ir

[1] Department of Statistics, University of Birjand, 97175/615 Birjand, Iran

[2] Faculty of Electrical and Computer Engineering, University of Birjand, 97175/615 Birjand, Iran

the input data into a higher dimensional feature space. As a result, the data can be separated linearly, and the classification will be performed with more accuracy.

Nowadays, pattern recognition plays an important role is industry. There are many industrial robots and other machines such as sorters that perform classification. Food industry is one the industries in which the accuracy of classification is very critical and important. For example, 1% false classification of agricultural products such as pistachio, olive and walnut can decrease the profit significantly. Thus, improving the accuracy of classifies such as SVM is necessary and important and will be applicable in today's modern industry.

After proposing SVM, extensive studies have been devoted to enhancing its performance using different strategies. Reducing the number of support vectors is one of these strategies. For example, the proposed approach in Downs et al. (2001) allows the recognition and elimination of unnecessary support vectors while leaving the solution unchanged. Also, v—support vector classification has been proposed that introduces a regularization parameter v to control the number of support vectors and margin errors (Gu and Sheng 2017). A robust SVM in Song et al. (2002) has been proposed that tries to solve the over-fitting problem when outliers exist in the training sets and reduce the number of support vectors. Feature extraction is another solution for enhancement of the SVM performance. In (Caoa et al. 2003), three approaches, namely kernel principal component analysis (KPCA), principal component analysis (PCA) and independent component analysis (ICA), have been compared for feature extraction in SVM. Feature normalization is another solution proposed in the literature (Steinwart et al. 2004; Bi et al. 2005). Modifying or changing the kernel function with the aim of enhancing SVM performance is another idea that has been studied extensively (Ye and Suganthan 2012; Zhang et al. 2004; Kuo et al. 2014; Izquierdo-Verdiguier et al. 2013; Moghaddam and Hamidzadeh 2016).

This paper presents a novel SVM using semi-parametric linear models. Regression analysis is a statistical approach for studying the relation between independent variables and a dependent variable. These methods are usually classified into two main methods, so-called parametric and nonparametric methods. Parametric models are very helpful for studying the relationship between variables. However, such methods may sometimes be applied at the risk of introducing modeling biases. In non-parametric models, no prior model structure is required. They can provide useful insight for further parametric fitting. However, they suffer from some drawbacks such as the curse of dimensionality, difficulty of interpretation, and lack of extrapolation capability. Therefore, semi-parametric linear models are more useful and are applied in many applications since both the parametric and nonparametric components can simultaneously exist in the model (Hesamian et al. 2017; Zarei et al. 2020).

This paper is organized as follows. Section 2 presents a brief explanation of SVM. Section 3 demonstrates the proposed SVM using semi-parametric linear regression. The results of some experiments on typical classification problems are illustrated in Sect. 4. Finally, Sect. 5 concludes the paper.

2 Support vector machine

2.1 Linear support vector machines

Without any loss of generality, the classification problem in SVM is demonstrated for two-class problems. The aim is to classify the two groups using a function which is obtained from available data. The main objective is to design a classifier performing well for unseen data. Consider the data in Fig. 1. It is obvious that there are many acceptable linear decision boundaries classifying the data, but there is only one that maximizes the margin (maximizes the distance between the classifier and the closest Lu data point of each group). This linear classifier is known as the optimal separating hyper-plane. Instinctively, we expect this boundary to generalize well in contrast to the other possible boundaries (Gunn 1998).

Consider the training dataset $\mathbf{x}_i \in R^d \;\; i = 1, 2, ..., N$ with a label $y_i \in \{-1, +1\}$ for all the training data and d is the

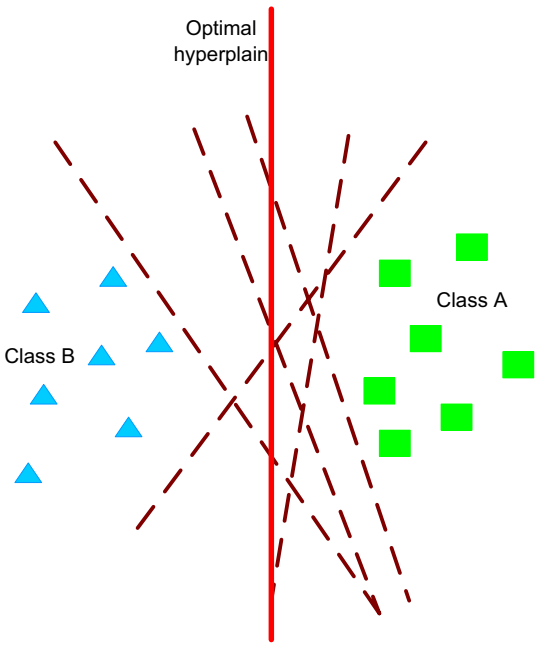

Fig. 1 Optimal separating hyper-plain

dimension of the problem. Consider the hyper-plain described by

$$\theta^T.\mathbf{x} + b = 0 \tag{1}$$

The classification problem is to obtain the hyper-plane such that $\boldsymbol{\theta}^T.\mathbf{x}_i + b \geq +1$ for positive class and $\boldsymbol{\theta}^T.\mathbf{x}_i + b \leq -1$ for negative class. According to Gunn 1998, in order to find the hyper-plane with the largest margin, the following minimizing problem should be solved:

$$\min_{\mathbf{w},b} J(\boldsymbol{\theta}) = \frac{\|\boldsymbol{\theta}\|^2}{2} \tag{2}$$

The constraint of this minimization problem is

$$y_i(\boldsymbol{\theta}^T.\mathbf{x}_i + b) - 1 \geq 0 \tag{3}$$

This quadratic programming (QP) problem can be solved by standard techniques such as Lagrange multipliers (Gutschoven and Verlinde 2000; Osuna et al. 1997) or intelligent optimization techniques such as particle swarm optimization (PSO) and genetic algorithm (GA) (Fateh and Khorashadizadeh 2012; Zadeh et al. 2016).

2.2 Nonlinear support vector machines and kernels

Real-life classification problems may be difficult to be solved by a linear SVM. Therefore, its extension to nonlinear decision boundary is inevitable. In nonlinear problems, the kernel function will map the input space into a high dimensional feature space by a nonlinear transformation. For example, consider Fig. 2. According to Cover theorem, input space can be converted into a new feature space in which the patterns can be linearly separable with high probability, if the dimensionality of the feature space is high enough (Haykin 1999). This nonlinear transformation is performed in implicit way through so-called kernel functions.

2.2.1 Inner-product kernels

In order to deal with nonlinear classification problems using SVM, a mapping $\varphi : R^n \rightarrow H$ is required. This mapping will transform the input data into the Euclidean space H which is a significantly higher dimensional. Now, the linear SVM is performed in the new space with dimension d. As a result, the training algorithm uses the data through dot product in H of the form $\varphi(\mathbf{x}_i)\varphi(\mathbf{x}_j)$. If the number of training vectors $\varphi(\mathbf{x}_i)$ is very large, then the calculation of the dot products will be time-consuming and computational. Moreover, φ is not known a priori. In this situation, Mercer theorem (Burges 1998) is used to replace $\varphi(\mathbf{x}_i)\varphi(\mathbf{x}_j)$ by a positive definite symmetric kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$. In other words, $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)\varphi(\mathbf{x}_j)$. To be more precise, kernel substitution paves the way for designing nonlinear methods from algorithms previously limited to dealing with linear separable input data (Campbell 2000). In addition, it prevents from.

the so-called challenge of dimension curse (Vapnik 1995). Some typical kernel functions are listed in Table 1.

2.3 Designing SVM based on Lagrange optimization

The optimization problem described by (2) and (3) is a typical quadratic programming problem, since the objective function $J(\boldsymbol{\theta})$ is quadratic and the constraints are linear. Based on this constrained optimization problem and considering the training set $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$ and the Lagrange multipliers $\{\alpha_i\}_{i=1}^N$, another problem called the dual problem can be formulated in the form of

$$\max\ Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j \tag{4}$$

subject to the constraints

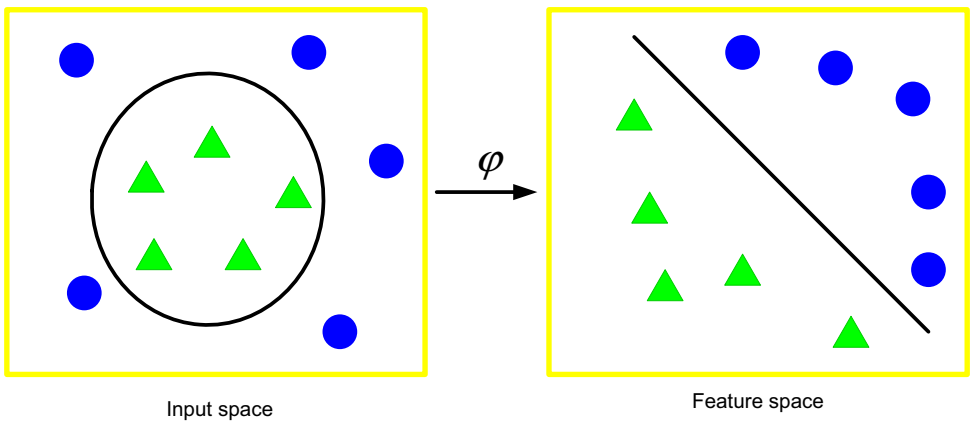

Fig. 2 Transforming the input data to the feature space using the nonlinear function φ

Table 1 Some typical kernels for nonlinear mapping in SVM

Gaussian (Radial-basis) kernel	$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2/2\sigma^2)$
Multi-layer perceptron (sigmoid)	$K(\mathbf{x}, \mathbf{x}_i) = \tanh(\beta_0 \mathbf{x}^T \mathbf{x}_i + \beta_1)$
Polynomial kernel	$K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i + 1)^d$

$$\sum_{i=1}^{N} \alpha_i d_i = 0 \tag{5}$$

$$\alpha_i \geq 0 \quad i = 1, 2, ..., N \tag{6}$$

For non-separable patterns, the constraint (6) should be modified as (Byun and Lee 2002)

$$0 \leq \alpha_i \leq C \qquad i = 1, 2, ..., N \tag{7}$$

where C is a user-defined positive parameter. This optimization problem can be solved by several methods mentioned in Byun and Lee (2002).

3 The proposed SVM based on semi-parametric linear regression

Consider the hyper plain that classifies the training dataset in the form of

$$\boldsymbol{\theta}^T.\mathbf{x}_i - b = 0 \qquad y_i \in \{-1, 1\} \quad i = 1, 2, ..., m \tag{8}$$

in which $y_i \in \{-1, 1\}$ $i = 1, 2, ..., m$ is the label of each sample $\mathbf{x}_i$. Suppose that $\boldsymbol{\theta}, \mathbf{x}_i \in \mathbb{R}^p$, i.e.,

$$\boldsymbol{\theta}^T = [\,\theta_1 \quad \theta_2 \quad ... \quad \theta_p\,]$$

and

$$\mathbf{x}_i^T = [\,x_{i1} \quad x_{i2} \quad ... \quad x_{ip}\,]$$

. The proposed method starts by rewriting (8) as

$$\boldsymbol{\theta}^T.\mathbf{x}_i - f(\mathbf{x}_i) = 0 \tag{9}$$

where $f(\mathbf{x}_i)$ is a term that will be calculated later. In fact, instead of obtaining a constant value for the parameter b in (8), we will make it dependent to the sample $\mathbf{x}_i$. In other words, $f(\mathbf{x}_i)$ is the term that results in the best classification performance with the smallest error. Since $f(\mathbf{x}_i)$ is unknown, we try to estimate it by

$$\hat{f}(\mathbf{x}_i) = \sum_{j=1}^{m} w_j(\mathbf{x}_i)(\boldsymbol{\theta}^T.\mathbf{x}_j) \tag{10}$$

$$w_j(\mathbf{x}_i) = \frac{K\left(\frac{\|\mathbf{x}_j - \mathbf{x}_i\|}{h}\right)}{\sum_{j=1}^{m} K\left(\frac{\|\mathbf{x}_j - \mathbf{x}_i\|}{h}\right)} \tag{11}$$

where $K(.)$ is the kernel function and h is the smoother parameter that is obtained by cross-validation measure (Campbell 2000). The kernel function has the following properties (Wasserman 2006):

$$\int K(u)du = 1, \quad \int uK(u)du = 0, \quad \int u^2 K(u)du = \sigma_k^2 > 0 \tag{12}$$

Some commonly used kernels are illustrated in Fig. 3.

Substitution of $\hat{f}(\mathbf{x}_i)$ from (10) into (9) results in:

$$\boldsymbol{\theta}^T.\mathbf{x}_i - \sum_{j=1}^{m} w_j(\mathbf{x}_i)(\boldsymbol{\theta}^T.\mathbf{x}_j) = 0 \tag{13}$$

In other words, we have $\boldsymbol{\theta}^T \mathbf{x}_i^* = 0$ in which

$$\mathbf{x}_i^* = \mathbf{x}_i - \sum_{j=1}^{m} w_j(\mathbf{x}_i)\mathbf{x}_j \tag{14}$$

Now, in order to obtain the optimal values for the vector $\boldsymbol{\theta}$ based on SVM that yields in the best classification performance, the following optimization problem should be solved:

$$L = \min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2 \ \ subject\ to\ \ y_i(\boldsymbol{\theta}^T.\mathbf{x}_i^*) - 1 \geq 0 \tag{15}$$

Using Lagrange optimization, the cost function (15) is rewritten in the form of

$$L_p = \min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i(\boldsymbol{\theta}^T.\mathbf{x}_i^*) - 1\right) \tag{16}$$

where α_i is called Lagrange multiplier. Therefore, differentiating L_p with respect to $\boldsymbol{\theta}$ and setting the result equal to zero $\left(i.e., \frac{\partial}{\partial \boldsymbol{\theta}} L_p = 0\right)$, we get:

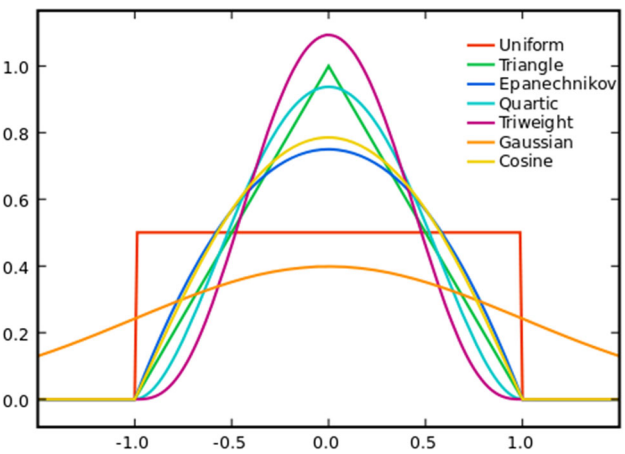

Fig. 3 Some commonly used kernels for calculation of weights (https://en.wikipedia.org/wiki/Kernel_(statistics))

$$\boldsymbol{\theta} = \sum_{i=1}^{m} \alpha_i y_i \, \mathbf{x}_i^* \tag{17}$$

Due to the duality theorem (Bertsekas 1995, we can reformulate the cost function (16) as

$$L_D = \sum_{i=1}^{m} \alpha_i \big(y_i(\boldsymbol{\theta}^T . \mathbf{x}_i^* + 0) - 1\big) - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i^{*T} . \mathbf{x}_j^* \right) \tag{18}$$

Substitution of (17) into (18) results in

$$L_D = \sum_{i=1}^{m} \alpha_i \left[y_i \left(\sum_{j=1}^{m} \alpha_j y_j \left(\mathbf{x}_i^{*T} . \mathbf{x}_j^* \right) \right) - 1 \right] - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i^{*T} . \mathbf{x}_j^* \right) \tag{19}$$

For classification problems in which the training set (input space) is not linearly separable, a nonlinear map is used to produce linearly separable data (feature space) (Haykin 2007). Suppose that $\{\varphi_j(\mathbf{x})\}_{j=1}^{m_1}$ is a set of nonlinear transformations from the input space to the feature space and m_1 is the dimension of the feature space. As a result, the hyper-plane acting as the decision surface is given by:

$$\sum_{j=1}^{m_1} \theta_j \varphi_j(\mathbf{x}) + b = 0 \tag{20}$$

Assuming $\theta_0 = b$ and $\varphi_0(\mathbf{x}) = 1$, the hyper-plane (20) can be simply converted to

$$\sum_{j=0}^{m_1} \theta_j \varphi_j(\mathbf{x}) = 0 \tag{21}$$

Adapting the optimal weight vector given in (17) to this new situation involving feature space where we now seek linear separability of the features yields in

$$\boldsymbol{\theta} = \sum_{i=1}^{m} \alpha_i y_i \, \varphi(\mathbf{x}_i^*) \tag{22}$$

As a result, the cost function of the dual problem given in (19) is rewritten as

$$L_D = \sup_{\alpha,\lambda} \left\{ \sum_{i=1}^{m} \left\{ \alpha_i \left(\sum_{j=1}^{m} y_i y_j \alpha_j \psi\left(\mathbf{x}_i^*, \mathbf{x}_j^* \right) \right) - \frac{1}{2} \sum_{i=1}^{m} \left\{ \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \psi\left(\mathbf{x}_i^*, \mathbf{x}_j^* \right) \right\} \right\} \right. \tag{23}$$

where $\psi\left(\mathbf{x}_i^*, \mathbf{x}_j^* \right) = \varphi(\mathbf{x}_i^*) \varphi(\mathbf{x}_j^*)$ is the inner-product kernel.

4 Experimental results

In order to investigate the performance of the proposed method in classification problems, some examples are presented. Then, Iris dataset will be classified using the proposed method (SP-SVM), and the results of some previous related works on Iris dataset will be presented for comparision.

Example 1 Consider the training data and the corresponding labels presented in Fig. 4. In other words, this training data can be described as given in Table 2.

The kernel used for the semi-parametric regression is a Gaussian kernel:

$$K\left(\frac{\left\| \mathbf{x}_j - \mathbf{x}_i \right\|}{h} \right) = \exp\left(- \frac{\left\| \mathbf{x}_j - \mathbf{x}_i \right\|^2}{h} \right) \tag{24}$$

and the polynomial kernel

$$\psi\left(\mathbf{x}_i^*, \mathbf{x}_j^* \right) = (\mathbf{x}_j^{*T} \mathbf{x}_i^* + 1) \tag{25}$$

has been selected for the nonlinear mapping in SVM. In this example, we have assumed that $h^{-1} = 0.005$. The weights calculated by the proposed method are $\theta_1 = 3.09602, \theta_2 = -2.64302$, and the predicted outputs (labels) are

$$y = \{0.999996, \; -1.52512, \; 4.05512, \; 1.53007, \; -0.999996, \; -3.53841, \; 2.04821\} \tag{26}$$

It is obvious that all the data have been correctly classified, since the signs of predicted values in (26) arethe same as the sign of values in the training data (the third row

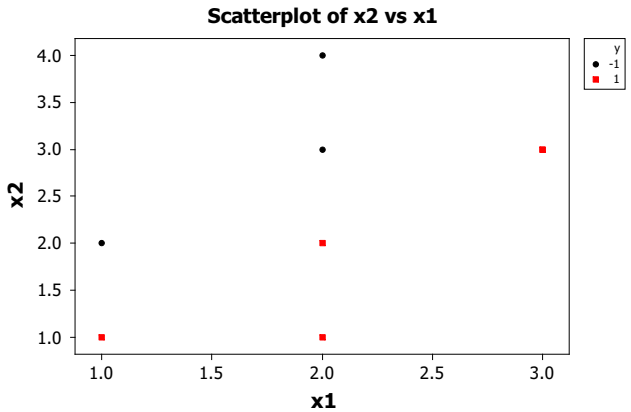

Fig. 4 Training data in Example 1

Table 2 Training data in Example 1

x_1	1	1	2	2	2	2	3
x_2	1	2	1	2	3	4	3
y	1	−1	1	1	−1	−1	1

in Table 2). In order to plot the decision boundary, one can use (14) to get

$$\mathbf{x}_i^* = \left(x_{i1}^*, x_{i2}^*\right) \qquad i = 1, 2, ..., 8$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{8} Exp\left[-0.005 \times \left(\left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2\right)\right] \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{8} Exp\left[-0.005 \times \left(\left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2\right)\right] \end{cases}$$

The values of $\left(x_{i1}^*, x_{i2}^*\right)$ are presented in Table 3. Moreover, the decision boundary in the $\left(x_{i1}^*, x_{i2}^*\right)$ plane is plotted in Fig. 5.

Thus, using the aforementioned optimal weights and values obtained for $\left(x_{i1}^*, x_{i2}^*\right)$ and the equation $\boldsymbol{\theta}^T \mathbf{x}_i^* = 0$, the decision boundary in the $\left(x_{i1}^*, x_{i2}^*\right)$ plane is given by

$$3.096 x_{i1}^* - 2.64 x_{i2}^* = 0 \qquad i = 1, 2, ..., 8$$

Example 2 Consider the training data and the corresponding labels presented in Fig. 6. In other words, this training data can be described as given in Table 4.

The kernel used for the semi-parametric regression is a the same as given in (24) with $h = 40$, and the polynomial kernel

$$\psi\left(\mathbf{x}_i^*, \mathbf{x}_j^*\right) = \left(\mathbf{x}_j^{*T} \mathbf{x}_i^* + 1\right)^2 \tag{27}$$

has been adopted for the nonlinear mapping in SVM. The weights calculated by the proposed method are $\theta_1 = -33.9996, \theta_2 = 21.9998, \quad \theta_3 = 9.99987, \theta_4 = -5.9996, \theta_5 = 2.0001$ and the predicted outputs (labels) are

$$y = \{0.99982,\ 6.99987,\ 0.9994,\ -0.99987\} \\ -10.9999,\ 9.9997,\ 0.9993, 0.9994\} \tag{28}$$

As it can be seen, the proposed method can correctly classify these data. In order to obtain the decision boundary, one can use (14) to get

Table 3 Transformed values in Example 1

x_{i1}^*	x_{i2}^*
− 14.5830	− 17.4607
− 14.7477	− 16.6983
− 13.6993	− 17.5816
− 13.8658	− 16.8212
− 13.8759	− 15.8758
− 13.7293	− 14.7436
− 12.8369	− 15.8121
− 12.8369	− 15.8121

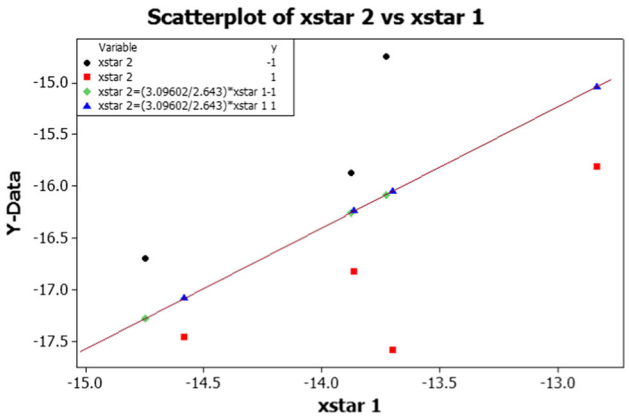

Fig. 5 Decision boundary of Example 1 in the $\left(x_{i1}^*, x_{i2}^*\right)$ plane

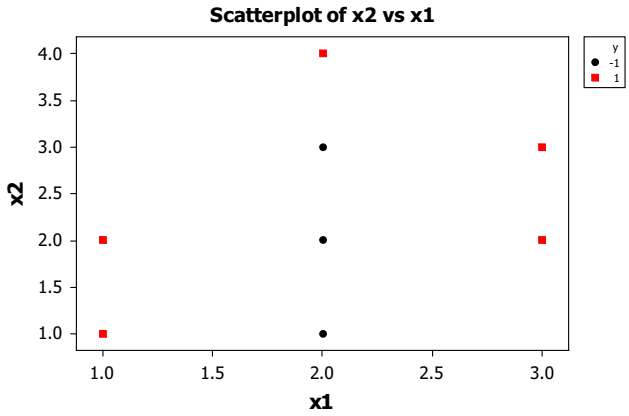

Fig. 6 Training data in Example 2

Table 4 Training data in Example 2

x_1	1	1	2	2	2	2	3	3
x_2	1	2	1	2	3	4	3	2
y	1	1	−1	−1	−1	1	1	1

$$\mathbf{x}_i^* = \left(x_{i1}^*, x_{i2}^*\right) \qquad i = 1, 2, ..., 8$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{8} 0.5 \times I\left[\begin{array}{l}(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2 \\ \leq 40\end{array}\right] \times x_{j1} \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{8} 0.5 \times I\left[\begin{array}{l}(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2 \\ \leq 40\end{array}\right] \times x_{j2} \end{cases}$$

$$I(A) = \begin{cases} 1 & A \\ 0 & A^c \end{cases}$$

Thus, using the aforementioned optimal weights and values obtained for $\left(x_{i1}^*, x_{i2}^*\right)$ and the equation $\boldsymbol{\theta}^T \mathbf{x}_i^* = 0$, the decision boundary in the $\left(x_{i1}^*, x_{i2}^*\right)$ plane is given by

$$-34x_{i1}^* + 22x_{i2}^* + 10\left(x_{i1}^2\right)^* - 6\left(x_{i2}^2\right)^* + 2(x_{i1}x_{i2})^* = 0 \quad i = 1, 2, ..., 8$$

Example 3 Consider the training data and the related classes presented in Fig. 7. This training data can be described as given in Table 5.

The kernel used for the semi-parametric regression is the same as given in (24) with $h = 40$, and the polynomial kernel $\psi\left(\mathbf{x}_i^*, \mathbf{x}_j^*\right) = \left(\mathbf{x}_j^{*T}\mathbf{x}_i^* + 1\right)^2$ has been adopted for feature transformation. The weights obtained by the proposed method are $\theta_1 = -28.33$, $\theta_2 = -5.77$, $\theta_3 = 22.1099$, $\theta_4 = 9.11057$, $\theta_5 = -19.5545$ and the predicted outputs (labels) are

$$y = \{15.22, -0.99, 0.99, -3.77, -39.55, -3.55, -0.99, 43.1, 20.22, 0.99\} \tag{29}$$

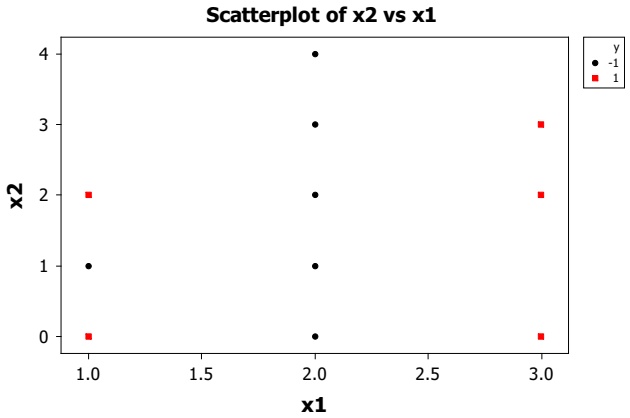

Fig. 7 Training data in Example 3

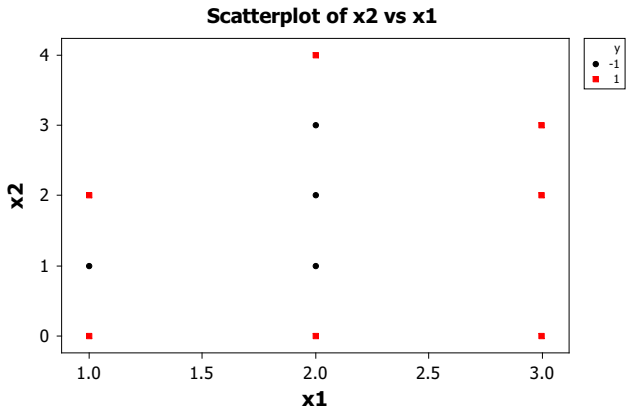

Fig. 8 Training data in Example 4

Table 5 Training data in Example 3

x_1	1	1	1	2	2	2	2	2	3	3	3
x_2	0	1	2	0	1	2	3	4	0	2	3
y	1	−1	1	−1	−1	−1	−1	−1	1	1	1

Table 6 Training data in Example 4

x_1	1	1	1	2	2	2	2	2	3	3	3
x_2	0	1	2	0	1	2	3	4	0	2	3
y	1	−1	1	1	−1	−1	−1	1	1	1	1

As it can be seen, the proposed method can correctly classify these data. In order to obtain the decision boundary, one can use (14) to get

$$\mathbf{x}_i^* = \left(x_{i1}^*, x_{i2}^*\right) \qquad i = 1, 2, ..., 11$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{11} 0.5 \times I\left[\begin{array}{l}\left(x_{i1}-x_{j1}\right)^2+\left(x_{i2}-x_{j2}\right)^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+\left(x_{i1}x_{i2}-x_{j1}x_{j2}\right)^2 \\ \leq 40\end{array}\right] \times x_{j1} \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{11} 0.5 \times I\left[\begin{array}{l}\left(x_{i1}-x_{j1}\right)^2+\left(x_{i2}-x_{j2}\right)^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+\left(x_{i1}x_{i2}-x_{j1}x_{j2}\right)^2 \\ \leq 40\end{array}\right] \times x_{j2} \end{cases}$$

$$I(A) = \begin{cases} 1 & A \\ 0 & A^c \end{cases}$$

Thus, using the aforementioned optimal weights and values obtained for $\left(x_{i1}^*, x_{i2}^*\right)$ and the equation $\boldsymbol{\theta}^T \mathbf{x}_i^* = 0$, the decision boundary in the $\left(x_{i1}^*, x_{i2}^*\right)$ plane is given by

$$\begin{aligned} &-28.33x_{i1}^* - 5.78x_{i2}^* \\ &+ 22.11\left(x_{i1}^2\right)^* + 9.11\left(x_{i2}^2\right)^* - 19.55\left(x_{i1}x_{i2}\right)^* \\ &= 0 \quad i = 1, ..., 11 \end{aligned}$$

Example 4 Consider the training data and the related classes presented in Fig. 8. This training data can be described as given in Table 6.

The kernel used for the semi-parametric regression is the same as given in (24) with $h = 40$, and the polynomial kernel $\psi\left(\mathbf{x}_i^*, \mathbf{x}_j^*\right) = \left(\mathbf{x}_j^{*T}\mathbf{x}_i^* + 1\right)^2$ has been used for feature transformation. The weights achieved by the proposed method are $\theta_1 = -6.15384, \theta_2 = -11.6923, \theta_3 = 5.84615, \theta_4 = 6.15383, \theta_5 = -4.76922$, and the predicted outputs (labels) are

$$\begin{aligned} y = \{&9.3, \ -0.99, \ 0.99, \ 3.61, \ -11.46, \ -2.076, \\ &-0.99, 12.3, 13.53, 1, 0.99\} \end{aligned} \tag{30}$$

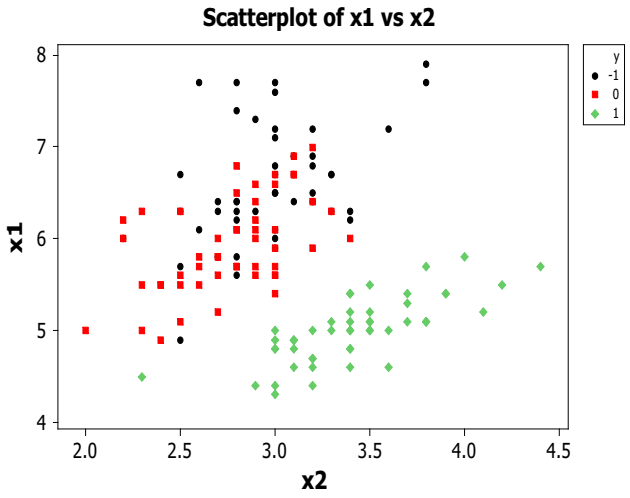

Fig. 9 Input features of Iris dataset in the (x_2, x_1) plane

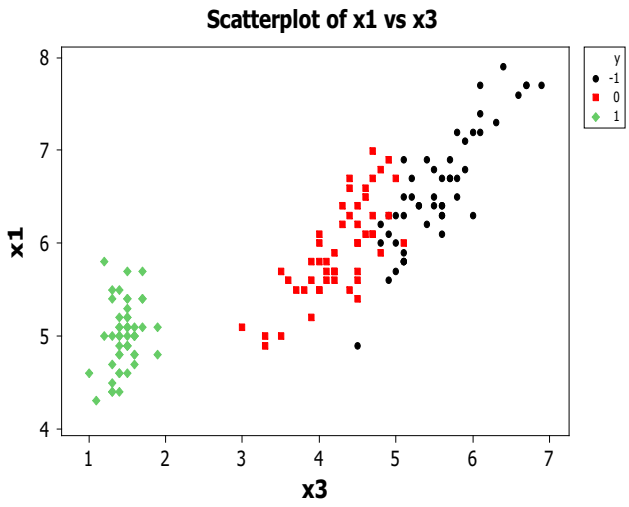

Fig. 10 Input features of Iris dataset in the (x_3, x_1) plane

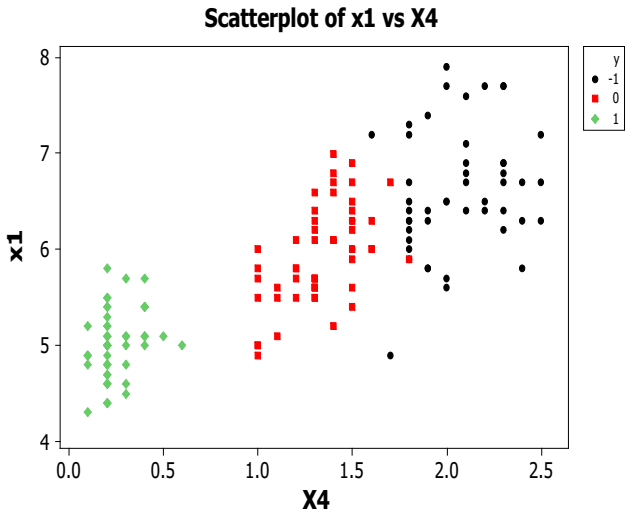

Fig. 11 Input features of Iris dataset in the $(x_4\,,\,x_1)$ plane

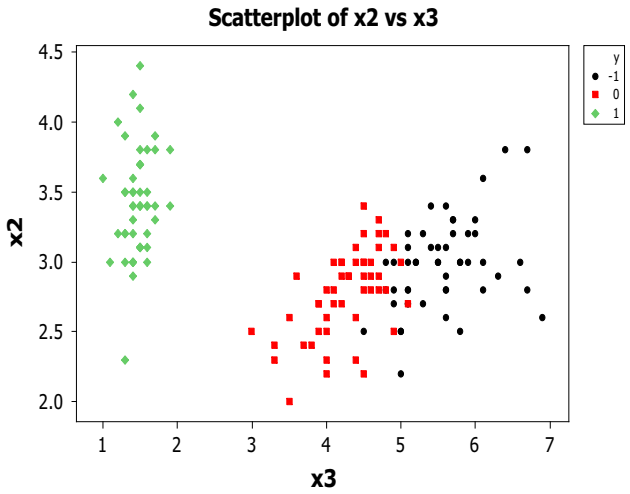

Fig. 12 Input features of Iris dataset in the $(x_3\,,\,x_2)$ plane

As it can be seen, the proposed method can correctly classify these data. In order to obtain the decision boundary, one can use (14) to get

$$\mathbf{x}_i^* = \left(x_{i1}^*\,,\,x_{i2}^*\right) \qquad i = 1, 2, ..., 11$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{11} 0.5 \times I\left[\begin{array}{l}\left(x_{i1}-x_{j1}\right)^2+\left(x_{i2}-x_{j2}\right)^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+\left(x_{i1}x_{i2}-x_{j1}x_{j2}\right)^2 \\ \leq 40\end{array}\right] \times x_{j1} \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{11} 0.5 \times I\left[\begin{array}{l}\left(x_{i1}-x_{j1}\right)^2+\left(x_{i2}-x_{j2}\right)^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2+\left(x_{i1}x_{i2}-x_{j1}x_{j2}\right)^2 \\ \leq 40\end{array}\right] \times x_{j2} \end{cases}$$

$$I(A) = \begin{cases} 1 & A \\ 0 & A^c \end{cases}$$

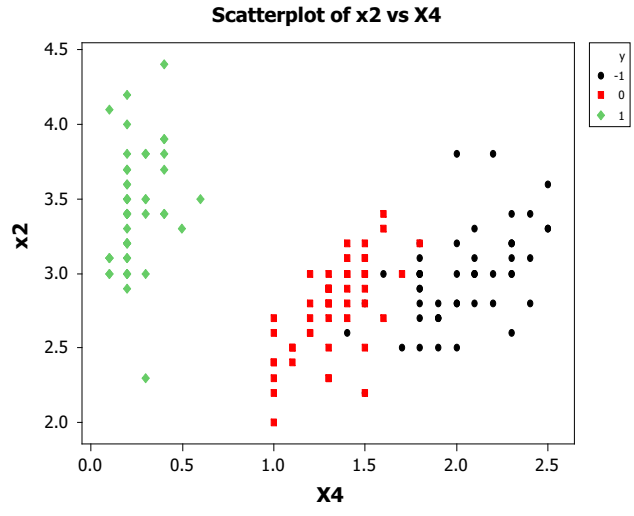

Fig. 13 Input features of Iris dataset in the $(x_4\,,\,x_2)$ plane

Thus, using the aforementioned optimal weights and values obtained for $\left(x_{i1}^*\,,\,x_{i2}^*\right)$ and the equation $\boldsymbol{\theta}^T\mathbf{x}_i^* = 0$, the decision boundary in the $\left(x_{i1}^*\,,\,x_{i2}^*\right)$ plane is given by

$$\begin{aligned} &-6.15x_{i1}^* - 11.698x_{i2}^* \\ &\quad + 5.85\left(x_{i1}^2\right)^* + 6.15\left(x_{i2}^2\right)^* + 4.77\left(x_{i1}x_{i2}\right)^* \\ &\quad = 0 \qquad i = 1, 2, ..., 11 \end{aligned}$$

Example 5 In order to test the performance of the proposed SVM on real data, Iris dataset from UCI Machine Learning Repository is used. Since there are 3 classes (+1, 0 and -1) in this dataset, we use the proposed method twice. Since there are 4 inputs in this dataset, it is impossible to plot them and distinguish the decision boundary. However, in order to decide which class should be separated in the first stage, consider Figs. 9, 10, 11, 12, 13, 14 in which some 2 dimensional plots of the input features are plotted. As seen in these figures, the green cluster with the desired output "+1" is completely separated from the other clusters. Therefore, in the first step, the data will be classified

into 2 groups: the green patterns are labeled by "+1," and the black and red patterns are labeled by "−1." In order to obtain the decision boundary, one can use (14) to obtain:

$$\mathbf{x}_i^* = \left(x_{i1}^*\ ,\ x_{i2}^*\ ,\ x_{i3}^*\ ,\ x_{i4}^*\right) \qquad i = 1, 2, ..., 150$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{11} 10 \times I \left[\begin{array}{l} \left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \left(x_{i3} - x_{j3}\right)^2 + \left(x_{i4} - x_{j4}\right)^2 + \left(x_{i1}^2 - x_{j1}^2\right)^2 + \left(x_{i2}^2 - x_{j2}^2\right)^2 \\ + \left(x_{i3}^2 - x_{j3}^2\right)^2 + \left(x_{i4}^2 - x_{j4}^2\right)^2 + \left(x_{i1}x_{i2} - x_{j1}x_{j2}\right)^2 + \left(x_{i1}x_{i3} - x_{j1}x_{j3}\right)^2 + \left(x_{i1}x_{i4} - x_{j1}x_{j4}\right)^2 \\ + \left(x_{i2}x_{i3} - x_{j2}x_{j3}\right)^2 + \left(x_{i2}x_{i4} - x_{j2}x_{j4}\right)^2 + \left(x_{i3}x_{i4} - x_{j3}x_{j4}\right)^2 \le 40 \end{array} \right] \times x_{j1} \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{11} 10 \times I \left[\begin{array}{l} \left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \left(x_{i3} - x_{j3}\right)^2 + \left(x_{i4} - x_{j4}\right)^2 + \left(x_{i1}^2 - x_{j1}^2\right)^2 + \left(x_{i2}^2 - x_{j2}^2\right)^2 \\ + \left(x_{i3}^2 - x_{j3}^2\right)^2 + \left(x_{i4}^2 - x_{j4}^2\right)^2 + \left(x_{i1}x_{i2} - x_{j1}x_{j2}\right)^2 + \left(x_{i1}x_{i3} - x_{j1}x_{j3}\right)^2 + \left(x_{i1}x_{i4} - x_{j1}x_{j4}\right)^2 \\ + \left(x_{i2}x_{i3} - x_{j2}x_{j3}\right)^2 + \left(x_{i2}x_{i4} - x_{j2}x_{j4}\right)^2 + \left(x_{i3}x_{i4} - x_{j3}x_{j4}\right)^2 \le 40 \end{array} \right] \times x_{j2} \\ x_{i3}^* = x_{i2} - \sum_{j=1}^{11} 10 \times I \left[\begin{array}{l} \left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \left(x_{i3} - x_{j3}\right)^2 + \left(x_{i4} - x_{j4}\right)^2 + \left(x_{i1}^2 - x_{j1}^2\right)^2 + \left(x_{i2}^2 - x_{j2}^2\right)^2 \\ + \left(x_{i3}^2 - x_{j3}^2\right)^2 + \left(x_{i4}^2 - x_{j4}^2\right)^2 + \left(x_{i1}x_{i2} - x_{j1}x_{j2}\right)^2 + \left(x_{i1}x_{i3} - x_{j1}x_{j3}\right)^2 + \left(x_{i1}x_{i4} - x_{j1}x_{j4}\right)^2 \\ + \left(x_{i2}x_{i3} - x_{j2}x_{j3}\right)^2 + \left(x_{i2}x_{i4} - x_{j2}x_{j4}\right)^2 + \left(x_{i3}x_{i4} - x_{j3}x_{j4}\right)^2 \le 40 \end{array} \right] \times x_{j3} \\ x_{i4}^* = x_{i2} - \sum_{j=1}^{11} 10 \times I \left[\begin{array}{l} \left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2 + \left(x_{i3} - x_{j3}\right)^2 + \left(x_{i4} - x_{j4}\right)^2 + \left(x_{i1}^2 - x_{j1}^2\right)^2 + \left(x_{i2}^2 - x_{j2}^2\right)^2 \\ + \left(x_{i3}^2 - x_{j3}^2\right)^2 + \left(x_{i4}^2 - x_{j4}^2\right)^2 + \left(x_{i1}x_{i2} - x_{j1}x_{j2}\right)^2 + \left(x_{i1}x_{i3} - x_{j1}x_{j3}\right)^2 + \left(x_{i1}x_{i4} - x_{j1}x_{j4}\right)^2 \\ + \left(x_{i2}x_{i3} - x_{j2}x_{j3}\right)^2 + \left(x_{i2}x_{i4} - x_{j2}x_{j4}\right)^2 + \left(x_{i3}x_{i4} - x_{j3}x_{j4}\right)^2 \le 40 \end{array} \right] \times x_{j4} \end{cases}$$

$$I(A) = \begin{cases} 1 & A \\ 0 & A^c \end{cases}$$

After running the optimization problem, the optimal weights are obtained and the optimal hyper-plane is given by

$$\begin{aligned} &-0.104x_{i1}^* - 0.042x_{i2}^* 0.0072x_{i3}^* + 0.0316x_{i4}^* + 0.0246\left(x_{i1}^2\right)^* \\ &-0.106\left(x_{i2}^2\right) + 0.227\left(x_{i3}^2\right)^* + 0.096\left(x_{i2}^2\right)^* - 0.0796(x_{i1}x_{i2})^* \\ &+0.0743(x_{i1}x_{i3})^* + 0.19(x_{i1}x_{i4}) + 0.1778(x_{i2}x_{i3})^* \\ &+0.146(x_{i2}x_{i4})^* + 0.188(x_{i3}x_{i4})^* = 0 \quad i = 1, 2, ..., 150 \end{aligned}$$

This hyper-plane can classify the patterns correctly without any error.

Now, a hyper-plane should be calculated to classify the red and black patterns. Therefore, the red group is labeled by "+1" and the black group is labeled by "−1." In order to obtain the decision boundary, one can use (14) to obtain:

$$\mathbf{x}_i^* = \left(x_{i1}^*, x_{i2}^*, x_{i3}^*, x_{i4}^*\right) \qquad i = 1, 2, ..., 150$$

$$\begin{cases} x_{i1}^* = x_{i1} - \sum_{j=1}^{11} 8 \times I \left[\begin{array}{l} (x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+(x_{i3}-x_{j3})^2+(x_{i4}-x_{j4})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2 \\ +\left(x_{i3}^2-x_{j3}^2\right)^2+\left(x_{i4}^2-x_{j4}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2+(x_{i1}x_{i3}-x_{j1}x_{j3})^2+(x_{i1}x_{i4}-x_{j1}x_{j4})^2 \\ +(x_{i2}x_{i3}-x_{j2}x_{j3})^2+(x_{i2}x_{i4}-x_{j2}x_{j4})^2+(x_{i3}x_{i4}-x_{j3}x_{j4})^2 \le 40 \end{array} \right] \times x_{j1} \\ x_{i2}^* = x_{i2} - \sum_{j=1}^{11} 8 \times I \left[\begin{array}{l} (x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+(x_{i3}-x_{j3})^2+(x_{i4}-x_{j4})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2 \\ +\left(x_{i3}^2-x_{j3}^2\right)^2+\left(x_{i4}^2-x_{j4}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2+(x_{i1}x_{i3}-x_{j1}x_{j3})^2+(x_{i1}x_{i4}-x_{j1}x_{j4})^2 \\ +(x_{i2}x_{i3}-x_{j2}x_{j3})^2+(x_{i2}x_{i4}-x_{j2}x_{j4})^2+(x_{i3}x_{i4}-x_{j3}x_{j4})^2 \le 40 \end{array} \right] \times x_{j2} \\ x_{i3}^* = x_{i2} - \sum_{j=1}^{11} 8 \times I \left[\begin{array}{l} (x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+(x_{i3}-x_{j3})^2+(x_{i4}-x_{j4})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2 \\ +\left(x_{i3}^2-x_{j3}^2\right)^2+\left(x_{i4}^2-x_{j4}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2+(x_{i1}x_{i3}-x_{j1}x_{j3})^2+(x_{i1}x_{i4}-x_{j1}x_{j4})^2 \\ +(x_{i2}x_{i3}-x_{j2}x_{j3})^2+(x_{i2}x_{i4}-x_{j2}x_{j4})^2+(x_{i3}x_{i4}-x_{j3}x_{j4})^2 \le 40 \end{array} \right] \times x_{j3} \\ x_{i4}^* = x_{i2} - \sum_{j=1}^{11} 8 \times I \left[\begin{array}{l} (x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+(x_{i3}-x_{j3})^2+(x_{i4}-x_{j4})^2+\left(x_{i1}^2-x_{j1}^2\right)^2+\left(x_{i2}^2-x_{j2}^2\right)^2 \\ +\left(x_{i3}^2-x_{j3}^2\right)^2+\left(x_{i4}^2-x_{j4}^2\right)^2+(x_{i1}x_{i2}-x_{j1}x_{j2})^2+(x_{i1}x_{i3}-x_{j1}x_{j3})^2+(x_{i1}x_{i4}-x_{j1}x_{j4})^2 \\ +(x_{i2}x_{i3}-x_{j2}x_{j3})^2+(x_{i2}x_{i4}-x_{j2}x_{j4})^2+(x_{i3}x_{i4}-x_{j3}x_{j4})^2 \le 40 \end{array} \right] \times x_{j4} \end{cases}$$

$$I(A) = \begin{cases} 1 & A \\ 0 & A^c \end{cases}$$

After performing the optimization procedure, the optimal weights are obtained and the optimal hyper-plane is given by:

$$\begin{aligned} &2.726x_{i1}^* + 15.808x_{i2}^* - 3.713x_{i3}^* - 7.835x_{i4}^* + 0.042\left(x_{i1}^2\right)^* \\ &+8.681\left(x_{i2}^2\right) + 1.687\left(x_{i3}^2\right)^* - 0.061\ \left(x_{i2}^2\right)^* - 7.84(x_{i1}x_{i2})^{**} \\ &+0.811(x_{i1}x_{i3})^* + 5.774(x_{i1}x_{i4}) - 3.98(x_{i2}x_{i3})^* \\ &-8.94(x_{i2}x_{i4})^* + 1.486(x_{i3}x_{i4})^* = 0 \quad i = 1, 2, ..., 150 \end{aligned}$$

Using these two hyper-planes, only 4 patterns of the 150 patterns in Iris dataset will be classified wrongly. Therefore, the accuracy of the proposed method is 97.33%.

In order to compare the performance of the proposed SVM with previous related works, consider the resulZahiri and Seyedin 2007ts in Siswantoro et al. (2016); Zahiri and Seyedin 2007; Tanveer et al. 2021). In (Siswantoro et al. 2016), a combination of Kalman filter and multi-layer perceptron (NN-LMKF) has been presented. In fact, a linear model based on Kalman filter has been used as a post-processing unit after the Multi-layer perceptron. In (Zahiri and Seyedin 2007), a swarm intelligence-based classifier (IPS) has been presented. In (Tanveer et al. 2021), some classification algorithms using K-nearest neighbor (KNN) have been presented. For example, the accuracy of K-nearest neighbor (KNN)-based weighted multi-class

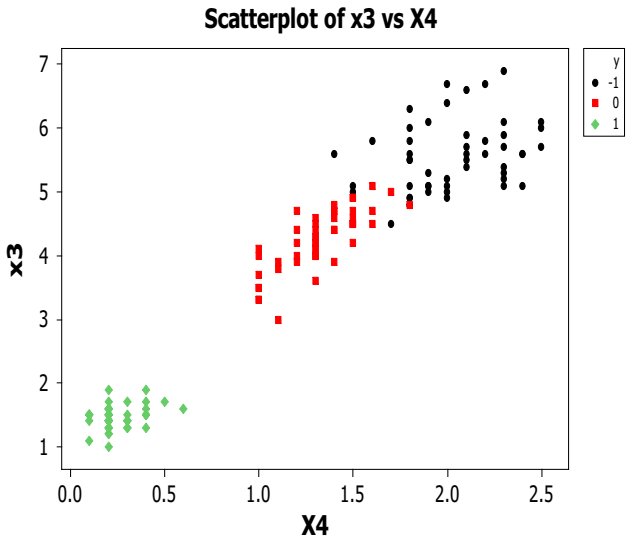

Fig. 14 Input features of Iris dataset in the (x_4, x_3) plane

Table 7 Comparision of Iris dataset classification accuracy of different algorithms

	Algorithm	Accuracy
Ref. (Siswantoro et al. 2016)	NN-LMKF	96.69%
Ref. (Zahiri and Seyedin 2007)	IPS	91.33%
Ref. (Tanveer et al. 2021)	Twin-KSVC	88.89%
Ref. (Tanveer et al. 2021)	LST-KSVC	86.66%
Ref. (Tanveer et al. 2021)	KWMTSVM	82.22%
Ref. (Tanveer et al. 2021)	LS-KWMTSVM	88.89%
Ref. (Wu et al. 2019)	Random Forest	63.4%
Proposed method	SP-SVM	**97.33%**

twin support vector machines (KWMTSVM), support vector classification–regression machine for K-class classification (K-SVCR) and twin multi-class classification support vector machines (twin-KSVC) has been reported in Tanveer et al. (2021). In (Wu et al. 2019), the results of a random forest classifier on Iris dataset have been presented. Table 7 compares the results of the aforementioned algorithms with the proposed method (SP-SVM). As shown in this paper, the proposed method outperforms the aforementioned algorithms.

5 Conclusion

In this paper, a new version of SVM has been proposed based on semi-parametric linear model. The similarity of the hyper-plane in SVM and parametric or semi-parametric models in statistics has been the main motivation of this paper. In other words, similar to semi-parametric regression model that the coefficients of the model are functions of the input data, a new version of SVM has been developed in this paper that parameters of the hyper-plane are functions of the input data. As a result, the proposed classifier is more flexible in comparison with the conventional SVM, since critical data can be classified in the correct set due to the fact that the parameters of the hyper-plane are not constant. The kernels used in data transformation are simple kernels such as polynomial kernels and the kernel used in the semi-parametric model is the Gaussian Kernel. The numerical results show that the proposed method can successfully be used in classification problems.

Acknowledgements Authors declare that the manuscript has not been submitted to more than one journal for simultaneous consideration and it has not been published previously (partly or in full). In addition, this study has not been split up into several parts to increase the quantity of submissions. No data have been fabricated or manipulated to support the conclusions. Moreover, no data, text or theories by others are presented as if they were our own. Proper acknowledgements to other works have been given.

Author contributions MGA developed the main ideas, generated the formulation of the algorithm and performed the experiments. SK provided the English text of the paper, edited the formulations and simulations, provided the reply letter and revised the manuscript. MM provided the references for literature review and proposed some experiments.

Funding No organization has funded this study.

Data availability The authors confirm that the data supporting the findings of this study are available within the article or its references.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human or animals rights Authors declare that there has been no human participant or animal in this research.

Informed consent Authors declare that there is no need for informed consent for this paper, since the results have been obtained using computer simulations.

References

Alajlan N, Bazi Y, Melgani F, Yager RR (2012) Fusion of supervised and unsupervised learning for improved classification of hyperspectral images. Inf Sci 217:39–55

Bashbaghi S, Granger E, Sabourin R, Bilodeau GA (2017) Dynamic ensembles of exemplar-svms for still-to-video face recognition. Pattern Recogn 69:61–81

Bertsekas DP (1995) Dynamic programming and optimal control. Athena scientific, Belmont, MA

Bi J, Chen Y, Wang JZ (2005) A sparse support vector machine approach to region-based image categorization. In: Proceedings

of the 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol 1, pp 1121–1128
Burges CC (1998) A tutorial on support vector machines for pattern recognition. In: Proceedings of international conference on data mining and knowledge discovery, vol 2, no 2, pp 121–167
Byun H, Lee SW (2002) Applications of support vector machines for pattern recognition: a survey. In: Lee S-W, Verri A (eds) Pattern recognition with support vector machines. Springer, Berlin, Heidelberg, pp 213–236. https://doi.org/10.1007/3-540-45665-1_17
Campbell C (2000) An introduction to kernel methods. In: Howlett RJ, Jain LC (eds) Radial basis function networks design and applications. Springer Verlag, Berlin
Caoa LJ, Chuab KS, Chongc WK, Leea HP, Gud QM (2003) A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. Neurocomputing 55:321–336
Christlein V, Bernecker D, Hönig F, Maier A, Angelopoulou E (2017) Writer identification using GMM supervectors and exemplar-SVMs. Pattern Recogn 63:258–267
Downs T, Gates KE, Masters A (2001) Exact simplification of support vector solutions. J Mach Learn Res 2:293–297
Dukart J, Mueller K, Barthel H, Villringer A, Sabri O, Schroeter ML, Alzheimer's Disease Neuroimaging Initiative (2013) Meta-analysis based SVM classification enables accurate detection of Alzheimer's disease across different clinical centers using FDG-PET and MRI. Psychiatry Res Neuroimaging 212(3):230–236
Fateh MM, Khorashadizadeh S (2012) Optimal robust voltage control of electrically driven robot manipulators. Nonlinear Dyn 70(2):1445–1458
Garcia MG, Rojo-Álvarez JL, Alonso-Atienza F, Martínez-Ramón M (2006) Support vector machines for robust channel estimation in OFDM. IEEE Signal Process Lett 13(7):397–400
Gu B, Sheng VS (2017) A robust regularization path algorithm for -support vector classification. IEEE Trans Neural Netw Learn Syst 28(5):1241–1248
Gunn SR (1998) Support vector machines for classification and regression, University of Southampton
Gutschoven B, Verlinde P (2000) Multi-modal identity verification using support vector machines (SVM). In: Proceedings of the third international conference on information fusion, pp 3–8
Haddi Z, Alami H, El Bari N, Tounsi M, Barhoumi H, Maaref A, Jaffrezic-Renault N, Bouchikhi BE (2013) Electronic nose and tongue combination for improved classification of Moroccan virgin olive oil profiles. Food Res Int 54(2):1488–1498
Haykin S (1999) Neural networks. Prentice Hall Inc, USA
Haykin S (2007) Neural networks: a comprehensive foundation. Prentice-Hall Inc
Hesamian G, Akbari MG, Asadollahi M (2017) Fuzzy semi-parametric partially linear model with fuzzy inputs and fuzzy outputs. Expert Syst Appl 71:230–239
https://en.wikipedia.org/wiki/Kernel_(statistics)
Izquierdo-Verdiguier E, Gomez-Chova L, Bruzzone L, Camps-Valls G (2013) Semisupervised kernel feature extraction for remote sensing image analysis. IEEE Trans Geosci Remote Sens 2(9):5567–5578
Kuo B, Ho H, Li C, Hung C, Taur J (2014) A Kernel-based Feature Selection Method for SVM with RBF Kernel for Hyperspectral Image Classification. IEEE J Select Top Appl Earth Oobserv Remote Sens 7(1):317–326
Liu Y, Wen K, Gao Q, Gao X, Nie F (2018) SVM based multi-label learning with missing labels for image annotation. Pattern Recogn 78:307–317
Moghaddam VH, Hamidzadeh J (2016) New Hermite orthogonal polynomial kernel and combined kernels in Support Vector Machine classifier. Pattern Recogn 60:921–935
Osuna E, Freund R, Girosi F (1997) Training support machines: an application to face detection. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 130–136
Ozer S, Chen C, Cirpan HA (2011) A set of new Chebyshev kernel functions for support vector machine pattern classification. Pattern Recogn 44:1435–1447
Roy A, Singha J, Devi SS, Laskar RH (2016) Impulse noise removal using SVM classification based fuzzy filter from gray scale images. Signal Process 128:262–273
Sapankevych, N. I., & Sankar, R. (2009). Time series prediction using support vector machines: a survey. *IEEE Computational Intelligence Magazine*, *4*(2).
Shah JH, Sharif M, Yasmin M, Fernandes SL (2017) Facial expressions classification and false label reduction using LDA and threefold SVM. Pattern Recogn Lett. https://doi.org/10.1016/j.patrec.2017.06.021
Siswantoro J, Prabuwono AS, Abdullah A, Idrus B (2016) A linear model based on Kalman filter for improving neural network classification performance. Expert Syst Appl 49:112–122
Solera-Urena R, García-Moral AI, Pelaez-Moreno C, Martinez-Ramon M, Diaz-de-Maria F (2012) Real-time robust automatic speech recognition using compact support vector machines. IEEE Trans Audio Speech Lang Process 20(4):1347
Song Q, Hu W, Xie W (2002) Robust support vector machine with bullet hole image classification. IEEE Trans Syst Man Cybern Part C (applications and Reviews) 32(4):440–448
Steinwart L, Schölkopf SB (2004) Sparseness of support vector machines–some asymptotically sharp bounds. Adv Neural Inf Process Syst 16
Subasi A (2013) Classification of EMG signals using PSO optimized SVM for diagnosis of neuromuscular disorders. Comput Biol Med 43(5):576–586
Tanveer M, Sharma A, Suganthan PN (2021) Least squares KNN-based weighted multiclass twin SVM. Neurocomputing 459:454–464
Vapnik V (1995) The nature of statistical learning theory. Springer
Wasserman L (2006) All of nonparametric statistics. Springer, New York
Wu Y, He J, Ji Y, Huang G, Yao H, Zhang P, Wen Xu, Guo M, Li Y (2019) Enhanced classification models for iris dataset. Procedia Comput Sci 162:946–954
Ye R, Suganthan PN (2012) A kernel-ensemble bagging support vector machine. In: 12th international conference on intelligent systems design and applications (ISDA), pp 847–852
Zadeh SMH, Khorashadizadeh S, Fateh MM, Hadadzarif M (2016) Optimal sliding mode control of a robot manipulator under uncertainty using PSO. Nonlinear Dyn 84(4):2227–2239
Zahiri SH, Seyedin SA (2007) Swarm intelligence based classifiers. J Franklin Inst 344(5):362–376
Zarei R, Akbari MG, Chachi J (2020) Modeling autoregressive fuzzy time series data based on semi-parametric methods. Soft Comput 24(10):7295–7304
Zhang L, Zhou W, Jiao L (2004) Wavelet support vector machine. IEEE Trans Syst Man Cybern Part B 34(1):34–39. https://doi.org/10.1109/TSMCB.2003.811113

Zhu F, Wei J (2017) Localization algorithm for large scale wireless sensor networks based on fast-SVM. Wireless Pers Commun 95(3):1859–1875