DESIGN AND ANALYSIS OF NON-PARALLEL SUPPORT VECTOR MACHINE ALGORITHMS FOR PATTERN CLASSIFICATION AND REGRESSION PROBLEMS

A Thesis submitted to Jawaharlal Nehru University
in partial fulfilment of the requirement
for the award of the degree of

Doctor of Philosophy
In
Computer Science

By

ANAGHA P

School of Computer and Systems Sciences
Jawaharlal Nehru University
New Delhi-110067, India

July 2022

Dedicated To

God , Teachers & My Parents

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI – 110067

Declaration

This is to certify that this thesis entitled “**Design and Analysis of Non-Parallel Support Vector Machine Algorithms for Pattern Classification and Regression Problems**” is being submitted to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, in partial fulfilment of the requirement for the award of the degree of **Doctor of Philosophy in Computer Science** is a record of bonafide research work carried out by me under the supervision of **Prof. T.V. Vijay Kumar** and Co-Supervision of **Prof. S. Balasundaram.**

The matter embodied in the thesis has not been submitted in part or full to any University or Institution for the award of any degree or diploma.

Anagha P

Mis. Anagha P
(Ph.D. Student)
Enrolment No:15/10/MT/011

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI – 110067

Certificate

This is to certify that this thesis entitled “**Design and Analysis of Non-Parallel Support Vector Machine Algorithms for Pattern Classification and Regression Problems**” submitted by **Ms. Anagha P** to the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, for the award of the degree of **Doctor of Philosophy in Computer Science** is a research work carried out by her under the supervision of **Prof. T.V. Vijay Kumar** and Co-Supervision of **Prof. S. Balasundaram.**

	18-07-2022
Prof. S. Balasundaram	**Prof. T.V. Vijay Kumar**
(Co-Supervisor)	**(Supervisor)**

18-07-2022

Prof. T.V. Vijay Kumar

(Dean)

Acknowledgements

This journey of my Ph.D. at Jawaharlal Nehru University has been a fascinating, enlightening, and beneficial experience. The work presented in this thesis would not have been possible without the help, support and suggestions of many people who were always there when I needed them the most. It is my pleasure to express my gratitude to them.

First and foremost, I would like to express my honest and sincere gratitude to my Ph.D. supervisors, Professor S. Balasundaram and Professor T.V. Vijay Kumar, School of Computer and Systems Sciences, Jawaharlal Nehru University, for their helpful advice, exceptional assistance, support, encouragement, and efforts. The Discussions with them have provided me with a wealth of knowledge and observing their work is a constant source of encouragement and inspiration for my research. Their gentle guidance, directions, suggestions, comments and enlightening ideas motivated me to give my best to this research work. I owe them a huge debt of gratitude and words are less to express how blessed I am to work with them. Successful completion of the present research effort has not been possible without them.

I wish to thank Professor T.V Vijay Kumar, Dean, School of Computer and Systems Sciences, Jawaharlal Nehru University, for providing the necessary research facilities.

I appreciate the support and cooperation of my labmates: Dr. Yogendra, Dr. Gagandeep, Dr. Subhash, Vivek, Sweta, Priyesh, Vikash, Sourv, Sumit, Shailender, Adarsh and Vikas for providing an excellent research environment. I would also like to thank all my friends, colleagues and administrative staff from the School of Computer and Systems Sciences and Central Library, Jawaharlal Nehru University.

Finally, I want to express my heartfelt gratitude, respect, and affection to my parents for their unwavering support, inspirational guidance, and endless love.

Anagha P

Anagha P

Abstract

This thesis uses support vector machines to solve pattern classification and regression issues in the context of machine learning (ML). The design, analysis, and implementation of new SVM variants for noisy datasets is the main contribution of this research. The goal is to create new ones with enhanced learning speed and better or comparable generalization performance on noisy samples. The efficacy of our proposed methodologies is demonstrated by experimental evidence on a variety of fascinating synthetic and real-world benchmark datasets.

The first chapter reviews prior work on SVM in the literature. The machine learning problem is introduced at the beginning of the chapter. ML is concerned with the study and development of learning algorithms that can learn from empirical data and produce prediction models with strong generalization capacity. Based on the availability of outputs, ML algorithms can be classified as supervised, unsupervised, or semi-supervised learning. The most prevalent tasks in the field of machine learning are classification and regression, hence supervised learning can be divided into two groups. SVM belongs to the category of supervised learning, which is the focus of this thesis. For classification and regression issues, SVM and neural networks are two of the most used machine learning techniques. The primary idea for training most ML algorithms, such as neural networks, is empirical risk minimization, which leads to overfitting. The structural risk minimization principle is utilized to solve this problem by determining the lowest upper bound on the projected risk. SVM is built on this same premise. Vapnik first proposed SVM for classification problems, and subsequently, by introducing the ε-insensitive loss, it was applied to regression problems, resulting in support vector regression (SVR). The twin SVM (TWSVM) and twin SVR (TSVR) are offered as ways to reduce the computational complexity of SVM and SVR training. Finally, the chapter concludes with a brief overview of variants of these models, such as SVM with pinball loss (Pin-SVM), ν-SVM, twin parametric margin SVM (TPMSVM), Least Squares TWSVM (LS-TWSVM), Twin Bounded SVM (TBSVM), twin SVM with pinball loss (Pin-TSVM), Least Squares SVR (LS-SVR), ν-SVR, ε-twin SVR (ε-TSVR), Lagrangian TSVR (LTSVR), twin parametric insensitive SVR (TPISVR), parametric insensitive non-parallel SVR (PINSVR).

The mathematical formulation and method of solution for well-known SVM variations for classification and regression problems, such as SVM, Pin-SVM, TWSVM, TBSVM, LS-

TWSVM, SVR and LS-SVR, are introduced in Chapter 2. SVM is a convex QPP that identifies the best separation hyperplane for a classification issue. Its objective function is made up of a regularization term that optimizes the margin between two classes and an empirical risk term that uses the hinge loss function to calculate the training error. The hyperplane must be at least one distance away from the training data points, according to the requirements. Pin-SVM is a classifier that is created by substituting the hinge loss function with a pinball loss function in the SVM to improve the SVM's noise insensitivity and resampling stability. TWSVM uses two smaller QPPs to locate two nonparallel hyperplanes, such as positive and negative hyperplanes. One of the QPP is solved to locate a positive hyperplane that is as close to class +1 data points as possible while being at least 1 distance from class -1 data points. Other QPPs are solved in the same way to discover a negative hyperplane that is as close to class -1 data points as possible and at least 1 distance from class +1 data points. TWSVM's learning pace is approximately four times faster as a result of SVM. The structural risk is implemented via a regularization term in the Twin Bounded Support Vector Machine, an upgraded version of TWSVM. This statistical learning theory technique can help you enhance your categorization performance. LS-TWSVM is generated by taking the square of the 2-norm of slack variables and transforming inequality constraints into equality constraints in TWSVM's primal QPPs, resulting in the solution of two systems of linear equations rather than a pair of QPPs as in TWSVM. SVR is a convex QPP whose solution determines a regression problem's regressor. Its objective function is made up of a regularization term that gauges regressor flatness and an empirical risk term that computes the training error using the ε-insensitive loss function, resulting in sparsity in the solution. Because of the limits, training data points must be crammed into the ε-tube as much as feasible. Instead of using an ε-insensitive loss, the LS-SVR uses a least square loss. In contrast to SVR, LS-SVR requires the solution of a system of linear equations rather than a QPP. In the spirit of TWSVM, TSVR solves two smaller QPPs rather to a single large QPP like that in SVR, resulting in a faster learning pace and high generalization ability.

In chapter 3, we presented a robust twin bounded support vector machine with pinball loss (Pin-TBSVM) for feature noise affected datasets. Pin-TBSVM is a non-parallel classifier where kernel generated surfaces were determined as the solutions of quadratic programming problems. Numerical tests on synthetic and benchmark datasets corrupted by noise are performed whose results are compared with a few popular classification learning algorithms. The comparative study confirms the effectiveness and suitability of our proposed methods.

In chapter 4, we present a $L_1 -$norm based twin bounded support vector machine with pinball loss for data categorization with the goal of obtaining an efficient robust learning model. Besides the benefit of less sensitivity to noise property of pinball loss, with the application of

$L_1 -$norm for within-class scatter minimization, the proposed method also enjoys robustness to outliers. As a novel approach of solving, by a simple reformulation of the primal problem considered, an equivalent pair of dual QPPs in *m* variables only is derived (L1-Pin-TBSVM), where *m* is the number of training vectors. In comparison with the twin bounded support vector machine (TBSVM), the duals of our proposed L1-Pin-TBSVM are free of inverse matrices and the non-linear duals can be obtained from their linear formulations directly by applying the kernel trick. Experiments on *Crossplanes* dataset where two or four outliers were introduced show that the proposed L1-Pin-TBSVM outperforms the other SVM methods in terms of accuracy, confirming its robustness to outliers. In addition, empirical findings based on a two-moon synthetic dataset and several benchmark datasets with varying levels of noise clearly demonstrate improved generalization ability of L1-Pin-TBSVM at comparable training cost which further confirms its effectiveness and suitability.

In chapter 5, with the aim of having the integrated merits of $L_1 -$norm and pinball loss in achieving enhanced robustness to outliers and bringing noise insensitivity to feature noise, we presented a novel, efficient $L_1 -$norm based non-parallel support vector machine classifier with pinball loss (L1-Pin-NPSVM) where its associated optimization problem minimizes the scatter loss and the misclassification error by $L_1 -$norm and pinball loss respectively. The dual formulation of the proposed method solves a pair of QPPs free of inverse kernel matrices. Our formulation allows a unified framework for the linear and nonlinear kernels. The effectiveness of L1-Pin-NPSVM is evaluated on synthetic and UCI benchmark datasets having outliers and/or being contaminated by noise. The results confirm its superiority in terms of robustness to outliers and noise insensitivity.

In chapter 6, we proposed a novel robust Huber SVR (HSVR) formulation in primal where the regressor is made as flat as possible by introducing the regularization term in L1-norm. Since the regularization term is non-smooth, it is proposed to replace it by smooth approximation functions and solve the problems by functional iterative method. Tests with both synthetic and real-world datasets confirm the suitability and effectiveness of the proposed robust model.

Chapter 7 summarises our research and identifies further research directions to be investigated.

List of Publications

This thesis is based on material from the following papers:

1. Prasad, S.C., Anagha, P., Balasundaram, S. (2022). Robust pinball twin bounded support vector machine for data classification. *Neural Processing Letters*. 10.1007/s11063-022-10930-6.
2. Puthiyottil, A., & Balasundaram. (2022). On Twin Bounded Support Vector Machine with Pinball Loss. In: Gupta D., Sambyo K., Prasad M., Agarwal S. (eds) Advanced Machine Intelligence and Signal Processing. Lecture Notes in Electrical Engineering, vol 858. Springer, Singapore. https://doi.org/10.1007/978-981-19-0840-8_13
3. Puthiyottil, A., Balasundaram, S., Meena, Y. (2020). L1-Norm Support Vector Regression in Primal Based on Huber Loss Function. In: Singh, P., Panigrahi, B., Suryadevara, N., Sharma, S., Singh, A. (eds) Proceedings of ICETIT 2019. Lecture Notes in Electrical Engineering, vol 605. Springer, Cham. https://doi.org/10.1007/978-3-030-30577-2_16

Contents

List of Figures

2 Support Vector Machine Techniques

4 Robust Twin Bounded Support Vector Machine with Pinball Loss for Data Classification

5 L1-norm Based Non-parallel Support Vector Machine Classifier with Pinball Loss

6 L1-norm Support Vector Regression in Primal Based on Huber Loss Function

List of Tables

2. Support Vector Machine Techniques

3. On Twin Bounded Support Vector Machine with Pinball Loss

4. Robust Twin Bounded Support Vector Machine with Pinball Loss for Data Classification

5. L1-norm Based Non-parallel Support Vector Machine Classifier with Pinball Loss

6. L1-norm Support Vector Regression in Primal Based on Huber Loss Function

List of Symbols

R:	The set of reals
R^n :	An n-dimensional space of reals
$\mathbf{x}$:	Column vector in R^n
$\mathbf{x}^t$:	Transpose of vector $\mathbf{x}$
$\mathbf{x}^t\,\mathbf{y}$:	Inner product of the vectors $\mathbf{x}$ and $\mathbf{y}$
$\mathbf{x}^t \perp \mathbf{y}$:	$\mathbf{x}$ is orthogonal to $\mathbf{y}$ component wise
$\\|\,\mathbf{x}\,\\|$:	L2-norm or 2-norm of vector $\mathbf{x}$
$\\|\,Q\,\\|$:	L2-norm or 2-norm of matrix Q
$\mathbf{x}_+$:	Vector $\mathbf{x}$ with all negative components set to zero
$(\mathbf{x})_i$:	i^{th} element of vector $\mathbf{x}$
$(\mathbf{x}_*)_i$:	$\begin{cases} 1 & \text{if } x_i > 0 \\ 0.5 & \text{if } x_i = 0, \\ 0 & \text{if } x_i < 0 \end{cases}$
$\mathbf{x} \geq \mathbf{0}$:	Vector $\mathbf{x}$ with all negative component
$diag(\mathbf{x})$:	A diagonal matrix having components of $\mathbf{x}$ as its diagonal elements
I :	Identity matrix of appropriate dimension
$\mathbf{e}$:	Column vector of ones of appropriate dimension
$K = K(M, N)$:	Kernel matrix K of size $m \times l$ with matrices $A \in R^{m\times n}$ and $N \in R^{n\times l}$
$\nabla f = (\partial f / \partial x_i, \ldots, \partial f / \partial x_n)^t$:	Gradient of real valued function f of the variables $\mathbf{x} = (x_1,\ldots,x_n)^t \in R^n$
$\nabla^2 f = (\partial^2 f / \partial x_i \partial x_j)_{i,j=1,\ldots,n}$:	Hessian Matrix

List of Abbreviations

AI:	Artificial Intelligence
aLS-SVM:	Asymmetric Least Squares Support Vector Machine
ANN:	Artificial Neural Network
ASVM:	Active Set Support Vector Machine
ERM:	Empirical Risk Minimization
GEPSVM	Generalized Eigenvalue Proximal Support Vector Machine
GHSVR	Huber Support Vector Regression by Generalized derivative
HSVR:	Huber Support Vector Regression
ITSVM:	Improved Twin Support Vector Machine
KKT:	Karush-Kuhn-Tucker
L1-LSTBSVM:	L1-norm based Least Squares Twin Bounded Support Vector Machine
L1-Pin-TBSVM:	L1-norm Pinball Twin Bounded Support Vector Machine
LS:	Least Squares
LS-SVM:	Least Squares Support Vector Machine
LS-SVR:	Least Squares Support Vector Machine for Regression
LSTPMSVM:	Least Squares Twin Parametric Margin Support Vector Machine
LSTSVM:	Least-Squares Twin Support Vector Machine
LSVM:	Lagrangian Support Vector Machine
LTSVR:	Lagrangian TSVR
ML:	Machine Learning
NPSVM:	Nonparallel Support Vector Machine
Par-v-SVM:	Parametric-margin v -Support Vector Machine
Pin-SVM:	Support Vector Machine with Pinball Loss
PINSVR:	Parametric-insensitive non-parallel Support Vector Regression

Pin-TSVM:	Twin Support Vector Machine with Pinball Loss
PSVM:	Proximal Support Vector Machine
PTSVM:	Projection Twin Support Vector Machine
PTSVR:	Primal Twin Support Vector Regression
QPP:	Quadratic Programming Problem
RPTSVM:	Projection TWSVM with Regularization
SHSVR1	Smooth Huber Support Vector Regression1
SHSVR2	Smooth Huber Support Vector Regression2
SLT:	Statistical Learning Theory
SMO:	Sequential Minimal Optimization
SOR:	Successive Over Relaxation
SRM:	Structural Risk Minimization
SSVM:	Smooth Support Vector Machine
SSVR:	Smooth Support Vector Regression
STSVR:	Smooth Twin Support Vector Regression
SVM:	Support Vector Machine
SVR:	Support Vector Regression
TBSVM:	Twin Bounded Support Vector Machine
TPISVR:	Twin Parametric Insensitive Support Vector Regression
TPMSVM:	Twin Parametric Margin Support Vector Machine
TSVR:	Twin Support Vector Regression
TWSVM:	Twin Support Vector Machine
WLTSVM:	Weighted Twin Support Vector Machine
ε-TSVR:	ε-Twin Support Vector Regression
γ-TSVM:	γ-Twin Support Vector Machine

Chapter 1

A Survey on Support Vector Machines (SVMs)

1.1 Introduction

In recent years Machine Learning, a subfield of Artificial Intelligence (AI) has appeared as one of the most fascinating scientific research areas in Computer Science. It helps computers to learn from experience without being explicitly programmed to enhance their performance. It focused on building an automated system that can learn by itself which leads the machine to make data-driven choices. Machine learning models improve with the use, and the more data they have access to, more the effective they become. Since machine learning has shown great success in building models for many real-world applications, it has been utilized in a wide range of domains such as Computer vision, Speech recognition, text understanding and GAME AI (Hull & Taylor, 1998; Kodratoff & Michalski., 2014).

There are four types of machine learning algorithms: supervised, unsupervised, semi-supervised, and reinforcement. The machine is taught by example in supervised learning methods. These models are made up of "input" and "output" data pairs, with the desired value labelled on the output. This category includes classification and regression tasks. In unsupervised learning models, the machine examines the input data, much of which is unlabelled and unstructured, and uses all relevant, accessible data to detect patterns and correlations. Unsupervised learning is similar to how humans see the world in many respects. To group things, we rely on intuition and experience. As we are exposed to more and more samples of something, our ability to categorize and identify it improves. The amount of data that is input and made available defines "experience" for machines. The third of four machine learning models is semi-supervised learning. All data would be structured and labelled before being entered into a system in an ideal world. However, because this is not possible, semi-supervised learning becomes a viable option when large amounts of unstructured data are available. Small amounts of labelled data are supplied to supplement unlabelled data sets in this approach. Essentially, the labelled data gives the system a head start and can enhance learning speed and accuracy significantly. A semi-supervised learning approach tells the machine to look for correlative qualities in the labelled data that could be applied to the unlabelled data. The fourth machine learning model is reinforcement learning. A set of permissible behaviours, rules, and potential end states are fed into the reinforcement learning model. Machines can learn by example when the algorithm's desired aim is fixed or binary. When the desired output is variable, however, the system must learn through experience and reward. The "reward" in reinforcement learning models is numerical and is coded into the algorithm as something that the system tries to acquire.

Machine learning's primary challenge is that it must perform well on new, previously unseen inputs, not merely those on which our model was trained. Generalization is the ability to perform well on previously unseen inputs. Machine learning approaches for developing classification models are widely used in a wide range of fields. In the training of classification and non-linear regression models, methods such as support vector machines (SVMs) and artificial neural networks (ANNs) have proved their high dependability.

Support Vector Machine (SVM) proposed by Vapnik et al. in 1992 is one of the most effective supervised learning techniques derived from statistical learning theory. Although it was originally developed as a binary classification method, its applications have extended to both multi-class classifications and regression problems, as well as function approximation. SVM has turned out to be one of the best classifiers for a wide range of situations, it has strong theoretical foundations and excellent empirical successes. After its introduction, SVM has outperformed most other systems in a range of practical applications within a few years. They proved to be excellent tools in the way of analyzing data and recognizing patterns as well as regression analysis. They combine characteristics from statistical learning theory, machine learning, and optimization theory, and one of their vital aspects is kernel functions. Due to its formulation based on a novel paradigm vested in the structural risk minimization induction principle (SRM principle), SVM has been the most promising machine learning method. Instead of minimizing an objective function based on the training samples such as mean square error (MSE), the SVM attempts to minimize a bound on the generalization error (i.e., the error made by the learning machine on test data not used during training). This is the distinction that allows the SVM to have strong generalization, i.e., greater prediction on data that has never been seen before. As a result, an SVM tends to perform well when applied to data outside the training set. It effectively avoids the local minimum and overfitting problem in classical machine learning methods such as neural networks (NNs), which perform Empirical Risk Minimization. Indeed, it has been reported that SVM-based approaches can significantly outperform competing methods in many applications. SVM achieves this advantage by focusing on the training examples that are most difficult to classify. These “borderline” training examples are called support vectors. SVM classification and regression problems have been mathematically proved to be optimization problems with a quadratic objective function and linear constraints, i.e., convex programming problems with a unique solution. The fact that SVM's solution is sparse, i.e., only some of the samples contribute to the determination of the decision function, is a clear advantage.

The SVM takes a collection of input data and predicts the classes to which each input should belong. The data points are represented as points in space in this model, with the data points of the different categories separated by a distinct and larger separation. This model generates a hyperplane (or a series of hyperplanes) for classification and regression analysis. SVM finds an ideal hyperplane for a linearly separable two-class problem that optimizes the separation between the two classes and hence lowers the generalization error. The input data are projected into another high-dimensional feature space for nonlinearly separable cases, making the data separable in that space. The data in the new feature space is then classified using SVM. The SVM classifier's performance and efficiency are determined by the best tuning parameters, which are commonly determined using the cross-validation method. Large sets of ideal parameters are also challenging to handle due to the training duration. Many different forms of SVMs are being developed to lower computational difficulties.

In several disciplines such as bioinformatics, handwriting recognition, and the stock market, Support Vector Machine(SVM) has been effectively applied to real-world data analysis issues, typically yielding better (or comparable) results than ANN. SVM and its extensions are one class of the most successful machine learning methods that outperform most other learning techniques in pattern recognition and regression estimation problems of practical importance, such as text classification (Joachims, 1998), interstellar object detection (Beaumont et al., 2011), combustion engine detection (Rychetsky et al., 1999), object detection (Pang et al., 2014), face detection (Osuna et al., 1997), financial time series forecasting (Mukherje et al, 1997; Tay & Cao, 2001; Kim, 2003), content-based image retrieval (Yildizer et al., 2012) handwritten digit recognition (Burges & Scholkopf, 1997; Cortes & Vapnik, 1995), object recognition (Papageorgiou et al., 1998), marketing (Ben-David & Lindenbaum, 1997), speech recognition (Schmidt & Herbert, 1996), medical diagnosis (Tarassenko et al., 1995), manufacturing yield estimation (Stoneking, 1999), and so on. Due to the wide range of real-world applications, it is considered one of the benchmarks in the field of statistical learning and machine learning.

The following are the key benefits of SVM:

- SVM maximizes the margin of the decision boundary using quadratic optimization techniques which find the optimal hyperplane. It finds a unique solution to a quadratic programming problem (QPP). Because optimization takes place in convex space, a global minimum is obtained, unlike with neural networks, which might become trapped at local minima. This form distinguishes SVM from Neural Networks, which provide many solutions based on local minima, making them untrustworthy across various samples.

- The spirit of the Structural Risk Minimization (SRM) principle is used in SVMs. It provides a trade-off between hypothesis space complexity (the VC dimension of approximation functions) and the quality of fitting the training data in a broad model of capacity control (empirical error). It employs the SRM principle, which aims to strike a balance between preventing overfitting and minimizing empirical risk.
- SVM can handle large feature spaces, therefore it's beneficial when there are more features than training samples.
- When we have no notion what data we are dealing with, SVMs come in handy. Only if unknown parameters are well-chosen, does SVM generalizes new data well (in Gaussian kernel). This also means that choosing a proper value can ensure good performance even if the input is biased. Works well with even unstructured and semi-structured data like text, Images, and trees.
- SVM's actual strength is the kernel technique. We can solve any complex problem with the right kernel function. Support vectors and the kernel function can be used to represent the resulting classifier.
- It handles high-dimensional data relatively well. As a result, it works on a higher dimension.
- In practice, SVM models have greater generalizability, and the risk of over-fitting is less.
- When classes are separable, SVM seems to be the best algorithm.
- Only the support vectors affect the hyperplane, hence outliers have less impact.
- They have two key advantages over newer algorithms like neural networks: higher speed and better performance with a limited number of samples (in the thousands)
- There are only a few parameters to optimize in SVM.
- They maximize the margin of the decision boundary using quadratic optimization techniques which find the optimal hyperplane.
- SVM is memory efficient

1.2 Support Vector Machine

By solving a QPP with linear inequality constraints, the SVM method determines a multidimensional hyperplane (Burges, 1998; Cristianini & Shawe-Taylor, 2000). It produces outstanding outcomes with a high degree of generalization. The fundamental disadvantage of SVM is that it has a high learning cost i.e., $o(m^3)$, where m is the number of training samples. This limits its applicability to minor and medium-sized issues. To overcome this flaw, Platt

(1998) proposed the Sequential Minimization Optimization (SMO) technique, which divides a large QPP into a sequence of smaller subproblems. Due to the usage of a single threshold value, the Platts model has a source of inefficiency. To address this issue, a new strategy was developed to improve the efficiency of the SMO algorithm (Keerthi et. al., 2001). For the problem of pattern classification and regression, SVMlight is a well-known version of Vapnik's SVM. It breaks down a major difficulty into a succession of smaller issues (Osuna et. al., 1997). This decomposition separates the data into working and non-working sets. It employs quick optimization methods based on the steepest feasible descent method for efficient working set selection (Joachims, 1999). The key benefit of this decomposition is that it can efficiently handle thousands of support vectors. Another novel method for classification and regression, known as LIBSVM, was proposed by Chang & Lin (2011). It is founded on the SMO concept, which was first suggested by Fan et al. (2005). It is now one of the most extensively used SVM implementations. Many SVM variations are supported by LIBSVM, including C-support vector classification (C-SVC) (Boser et al., 1992; Cortes and Vapnik, 1995), *ν*-support vector classification (*ν* -SVC) (Scholkopf et al., 2000), one class SVM (Scholkopf et al., 2001), epsilon SVR (ε-SVR) (Scholkopf et al., 2000), and *ν*-support vector regression (Scholkopf et al. 2001). A novel form of SVMTorch proposed (Collobert & Bengio, 2001), which is similar to SVMLight proposed by Joachims (1999) for classification tasks, is introduced by introducing the idea of a decomposition algorithm. The key advantage is that it can handle large-scale problems quickly.

1.2.1 **Classification**

Simplified dual and successive overrelaxation approaches to solve this dual problem have been developed for a basic reformulation of SVM derived by adding one-half of the square of the bias term in the regularization term (Mangasarian & Musicant, 1999). The linear equality restriction is removed in dual with this formulation, allowing one variable to be changed at a time. As a result, a very simple updating rule emerges. Mangasarian and Musicant (2001a) looked into the simplified dual formulation of 2-norm SVM and how to solve it using an active set technique. The 2-norm SVM is a modified SVM created by doubling the bias factor in the regularization term and measuring training loss in squared not linear. The 2-norm approach has the advantage of resulting in a simpler positive definite dual problem with simply non-negativity restrictions. They limit themselves to linear SVM and describe their solution using an active set technique that only requires matrix inversions that scale with the number of features rather than the size of the training dataset. For low-dimensional feature spaces, this results in a highly quick approach.

An unconstrained optimization (implicit Lagrangian) issue for the dual of 2-norm linear SVM is proposed, as well as a simple linearly convergent iterative algorithm for solving it termed Lagrangian SVM (LSVM) is proposed by (Mangasarian & Musicant, 2001b). Because it involves the inversion of a matrix of order equal to the dimensions of the input space plus one at the start of the iteration, the LSVM approach can handle small to moderate datasets. For non-linear kernel classification, the LSVM technique does not scale up to very big problems. For solving the unconstrained optimization problem for dual of 2-norm SVM, (Fung & Mangasarian, 2003) proposes a finite Newton approach that can handle huge datasets for both linear and non-linear classifications. In a novel SVM formulation termed smooth SVM (SSVM) and a fast Newton algorithm for solving it, a new unconstrained optimization problem is considered, and a smoothing technique is applied (Lee & Mangasarian, 2001). The reader is referred to for a study of implicit Lagrangian formulation related to the dual of 2-norm SVM and Newton method of solving it (Mangasarian & Solodov, 1993, Fung & Mangasarian, 2003). In (Fung & Mangasarian, 2001), a new approach known as the proximal support vector machine (PSVM) classifier is introduced, in which points are classified based on their proximity to one of two parallel planes that are pushed apart as far as feasible. Unlike SVM, which requires the solution of a QPP, PSVM just requires the solution of a system of linear equations. In order to extend the PSVM to fuzzy models, a fuzzy PSVM is presented (Jaydeva et al., 2004), in which each training point is assigned a fuzzy membership for approximating the two parallel planes that are kept as far apart as possible, and new points are classified based on their proximity to one of the two parallel planes. (Fung & Mangasarian, 2002) investigates a new method known as incremental SVM classification, which is capable of updating an existing linear classifier by both retiring old data and adding new data.

In (Sukeyns & Vandewalle, 1999) a basic reformulation of SVM called least squares SVM (LS-SVM) is introduced, which involves equality constraints rather than inequality constraints. Instead of using QPP as in SVM, LS-SVM solves a system of linear equations. LS-SVM, on the other hand, is less robust, and its solution is not sparse, meaning that every training point contributes to the final result. Weighted LS-SVM is presented to overcome the sparsity problem in LS-SVM (Suykens et al., 2002a). Furthermore, for achieving sparsity in LS-SVM, (Cawley & Talbot, 2002) suggests enhanced sparse LS-SVM.

In (Schollkopf et al., 2000), a new extension of SVM called v-SVM is proposed, in which parameter v is employed in the objective function instead of the regularization parameter. The advantage of the v-SVM model is that it uses parameter v to adjust the number of support vectors and training mistakes. When the data contains heteroscedastic error structure, i.e., noise greatly relies on data location, a variation of v-SVM based on the parametric margin model, entitled par- ν -SVM, is proposed (Hao et al., 2008). In (Wang et al., 2014) we discuss a new fast technique based on proximal parametric margin-based support vector machines.

Unlike PSVM, which seeks proximal parallel planes, a multisurface proximal SVM proposed by Mangasarian & Wild (2006) determines non-parallel proximal planes by dropping the parallel condition on planes and requiring that data points of each class be proximal to one of them while being as far away from the other plane as possible. This leads to the solution of two generalized eigenvalue problems, the outcomes of which are eigenvectors that define the non-parallel planes.

Instead of solving two generalized eigenvalue problems as in (Mangasarian & Wild, 2006), Jaydeva et al. (2007) proposed a new nonparallel classifier called twin SVM (TWSVM), in which two nonparallel planes are constructed with one plane being close to data points of one class and the other being as far away as possible from data points of the other class. However, rather than solving a single huge QPP as in SVM, this results in TWSVM solving two smaller QPPs, which makes TWSVM's training speed four times faster. To include the SRM principle into TWSVM, Shao et al. (2011) suggested a twin bounded SVM (TBSVM), which solves dual problems using the SOR technique to speed up training and appears to be capable of dealing with large scale problems. In contrast to TBSVM, an improved twin SVM (ITSVM) suggested in (Tian et al., 2014) does not require the computation of inverse matrices before training, and linear ITSVM can be simply extended to nonlinear cases. Tian et al. (2013a) suggested a nonparallel support vector machine (NPSVM) that uses an ε-insensitive loss instead of the quadratic loss used in TWSVM, TBSVM, or ITSVM to solve the sparsity problem. In order to gain the benefit of solving two systems of linear equations, least-squares twin SVM (LSTSVM) was introduced by Kumar & Gopal (2009), in which equality constraints are assumed in two modified primal QPPs of TWSVM rather than solving a pair of dual QPPs with inequality constraints.

Peng (2010c) presented a ν-twin support vector machine by adding new variables and parameters ν that govern the number of support vectors and margin errors (ν-TSVM). A unique twin parametric margin support vector machine (TPMSVM) is suggested in the spirit of TWSVM and analogous to par- ν-SVM (Peng, 2011a). Shao et al (2013b), inspired by the work of TPMSVM, proposed least-squares TPMSVM (LSTPMSVM), which solves two fundamental problems and produces two systems of linear equations. Ye et al. (2012) introduced a novel weighted TWSVM with local information (WLTSVM) that exploits the similarity information within samples and considers only the penalty parameter in order to achieve higher or equivalent classification accuracy with low computational cost than TWSVM.

In (Chen et al., 2011), the projection twin support vector machine (PTSVM) is a novel binary classifier that sets projection directions for each class, ensuring that projected samples of one class are well segregated from those of other classes in their respective space. Shao et al. (2012) introduced a fast least square projection TWSVM (LSPTSVM), in which a regularization term was included, and primal problems were handled by solving two systems of linear equations rather than the customary two dual problems. To overcome the PTSVM singularity problem, a formulation called projection TWSVM with regularization (RPTSVM) was developed by adding a regularization term to PTSVM (Shao et al., 2013a).

An SVM classifier with pinball loss (Pin-SVM) was recently presented in order to tackle the noise sensitivity and instability to resampling difficulties in SVM (Huang et al., 2014b). Pin-SVM is comparable to SVM in terms of computing complexity, noise insensitivity, and resampling stability. In (Huang et al., 2014a), the asymmetric least squares support vector machine (aLS-SVM) is investigated using squared pinball loss for classification. Xu et al. (2016) suggested a novel twin support vector machine with pinball loss based on the TPMSVM and pinball loss experiments (Pin-TSVM). When compared to Pin-SVM, Pin-TSVM is faster and achieves better results.

1.2.2 **Regression**

Regression is a statistical model that attempts to find the underlying mathematical relationship between one dependent variable and a series of other changing variables, known as independent variables. Legendre and Gauss created the least square approach in the early nineteenth century, and it is one of the most extensively used techniques for regression problems. By reducing the sum of the squares of the vertical deviations from each data point to the line, it calculates the best-fit line for the observed data. When the input qualities are highly correlated, the least square approach suffers from autocorrelation and overfitting, and the result is an unstable solution. To eliminate the possibility of overfitting, Hoerl (1962) presented a novel method called ridge regression. By adding a regularisation element to the objective function, it improves generalization across unknown data, which is an improvement over the least square technique. This problem's solution is found by solving a system of linear equations. A dual version of ridge regression (Saunders et al., 1998; Gammerman et al., 2004) enables us to do non-linear regression by employing kernel functions to generate a linear regression function in a high-dimensional feature space. Kernel functions are used to represent the dot product in a feature space created by the ANOVA decomposition approach. See (Saunders et al., 1998) for an extension to a fuzzy regression model with crisp inputs and Gaussian fuzzy outputs (Hong et al., 2004).

SVMs were invented in the first place to solve classification challenges. However, there is a different type of challenge known as regression problems. Vapnik et al. (1997) proposed support vector regression (SVR) as an extension of support vector machine (SVM), which has demonstrated good generalization performance for time series prediction (Mukherjee et al., 1997; Muller et al., 1999; Tay & Cao, 2001) and function approximation problems in fields such as civil engineering (Dibike et al., 2000), and drug discovery (Demiriz et al., 2001). SVR can be formulated as a quadratic programming problem (QPP) having a global optimal solution. All of the main properties of the maximal margin algorithm, such as convexity, duality, sparseness, and kernel, are preserved in SVR. SVR introduces an ε-insensitive loss function and accepts training data as input, producing a model that ignores any training data that is close (within a threshold ε) to the model prediction, similar to SVM for classification. It is well-known that the error of misfit in SVR is measured using $\varepsilon -$insensitive function introduced by Vapnik (2000) as the loss function. This function has the property that only samples falling outside of the $\varepsilon -$tube around the regression function are considered for computing the misfit error and these samples are the support vectors.

Smoothing approaches have been effectively utilized to solve a variety of mathematical programming issues, including support vector machine pattern categorization (Lee & Mangasarian, 2001). A smooth SVR (SSVR) was proposed by Lee et al. (2005) as an unconstrained smooth formulation of ε-insensitive SVR achieved by approximating the ε-insensitive loss function with a smooth function. In (Balasundaram & Singh, 2010), an ε-insensitive support vector regression is given as an unconstrained optimization problem whose solution is obtained by solving a system of linear equations using the Newton technique. For classification problems, (Mangasarian & Musicant, 2001b) proposes a novel method called Lagrangian SVM. Lagrangian ε-insensitive SVR formulation is given in based on Lagrangian SVM for regression problems (Balasundaram & Kapil, 2010). The main advantage of this method is that, rather than solving a quadratic optimization problem, this approach obtains its solution by taking the inverse of a matrix of order equal to the number of input samples at the start of the iteration. Balasundaram & Kapil (2011) developed an implicit Lagrangian SVR formulation whose solution is determined by a set of linear equations based on implicit Lagrangian SVM (Fung & Mangasarian, 2003) for classification. The active set SVM (ASVM) in (Mangasarian & Musicant, 2001a) method proposed for classification problems is extended to regression problems (Musicant & Fienberg, 2004). For an overview of SVM training implementation strategies see these papers (Abe, 2005; Schoelkopf & Smola, 2002; Steinwart & Christmann; 2008; Shawe-Taylor & Sun, 2011). The interested reader is directed to (Ji & Sun, 2013; Sun, 2011; Xie & Sun, 2012) an interesting and challenging work on multitask learning and multiview learning methods as extensions of kernel-based approaches.

In the spirit of TWSVM (Jayadeva et al., 2007), Peng (2010a) developed twin support vector regression (TSVR) for function approximation and regression. TSVR generates a pair of $\boldsymbol{\varepsilon}$ $-$insensitive down-bound and up-bound functions such that (i) both the functions lie as close as possible to the training data; (ii) all the training data is required to lie above the down-bound function but below the up-bound function; (iii) for each training data, its distance from the down-bound function and similarly from the up-bound function should be at least ε. This strategy results in solving a pair of smaller sized QPPs than solving a single QPP of large size as in the standard support vector regression (SVR) learning model (Peng 2010a). The advantage of this methodology is that the time complexity of TSVR becomes significantly smaller than SVR (Peng, 2010a). Primal TSVR(PTSVR) is formulated (Peng, 2010b) to increase the sparsity of TSVR by optimizing the pair of bound functions in the primal space using a quadratic function for approximating the non-differentiable loss function.

PTSVR must solve a set of linear equations to identify the optimal solution in the primal space, r esulting in increased learning speed. Peng (2012) introduced an efficient twin parametric insensitive SVR (TPISVR) that determines a pair of parametric insensitive down- and up-bound functions that are solved by two lower-sized QPPs. TPISVR is faster than SVR and works well with heteroscedastic noise. Yang et al. (2016) proposed a new parametric-insensitive non-parallel SVR (PINSVR) that determines a pair of nonparallel proximal functions with a pair of different parametric-insensitive nonparallel proximal functions solved by two smaller sized QPPs, making it suitable for heteroscedastic noise structure. PINSVR was found to have equivalent regression performance to TPISVR, as well as improved bound estimations (Yang et al., 2016).

A new formulation of TSVR is provided in (Zhong et al., 2012) based on solving pair of linear programming problems instead of quadratic programming problems, with the use of 1-norm distance instead of the square of 2-norm, resulting in higher generalization performance. In (Chen et al., 2012), a new SVR called smooth twin support vector regression (STSVR) was proposed, whose solution was obtained using the Newton-Armijo algorithm by slightly changing the formulation of TSVR as a pair of strongly convex unconstrained minimization problems in primal and employing smooth technique (Lee & Mangasarian, 2001b; Mangasarian, 2002). Lagrangian TSVR (LTSVR) has been proposed as another TSVR version (Balasundaram & Tanveer, 2013a). Motivated by the work of implicit Lagrangian SVM for classification (Fung & Mangasarian, 2003) and STSVR, Balasundaram & Tanveer (2013b), introduce implicit Lagrangian smooth twin SVR (LSTSVR), whose solution is achieved using the Newton-Armijo algorithm (Lee and Mangasarian, 2001; Mangasarian, 2002).

The fact that $\boldsymbol{\varepsilon}$ $-$SVR (Scholkopf & Smola, 2002) is based on the SRM concept and contains both a regularisation term and an empirical risk term is a significant advantage. TSVR, on the other hand, employs the empirical risk minimization (ERM) principle. The $\boldsymbol{\varepsilon}-$twin support vector regression regressor is based on the SRM principle presented in (Shao et al., 2013c), and it finds two proximal functions by solving two smaller sized QPPs using the successive overrelaxation (SOR) technique. In comparison to TSVR, $\boldsymbol{\varepsilon}-$TSVR has demonstrated a significant improvement in generalization performance while requiring less training time. Even if the data points are in different locations in the input space, all training data points in TSVR get the same penalties. Because of their varying effects on the bound functions, it is fairer to assign distinct penalties to data points. Based on this approach, a weighted TSVR (Xu & Wang, 2012) is developed, in which different penalties are assigned to samples based on their locations in order to address the over-fitting problem and improve generalization ability.

A robust SVR is proposed by Chuang et al. (2002) to overcome the influence of outliers in the training dataset, in which a robust cost function, namely the tanh-estimator, is used instead of the $\boldsymbol{\varepsilon}$-insensitive loss function. Chen et al. (2017a) suggested a least absolute deviation based robust SVR, motivated by the great robustness of least absolute deviation to outliers. Huang et al. (2014a) proposed an asymmetric *v*-tube SVR as an extension of *v*-SVR (Scholkopf et al., 2000), in which an asymmetric loss is used to deal with outliers and parameters *v* and *p* control the fraction of training data points above and below the tube to improve the flexibility of the insensitive tube location.

The construction of robust regression models for noisy data samples or data samples having outliers is a challenging research problem. The loss function plays an important role in obtaining a robust regression model. The popular loss functions used in the literature are the (i). quadratic; (ii). absolute value; (iii). $\boldsymbol{\varepsilon}-$insensitive functions. The quadratic function is smooth but is sensitive to samples having a large error of deviation. Though the absolute value and $\boldsymbol{\varepsilon}-$insensitive functions are less sensitive to noise, they are only continuous. Huber suggested a hybrid yet smooth loss function in which he combined robust linear treatment for large errors with quadratic treatment for minor errors (Huber, 1981). It has sparked a lot of discussion in the literature because, unlike the quadratic function, outliers with significant deviations are not overemphasized, and training points near the prediction function are penalized less than with 1 -norm. See regression models with Huber loss function in dual solved by Newton method for more information (Smola, 1998; Madsen & Nielsen, 1990). The robust Huber error function is treated as a convex QPP in the primal space in conjunction with SVR (Zhu et al., 2008; Mangasarian & Musicant, 2000). The fundamental benefit of Huber is that it may be used to solve problems involving data that has been contaminated by noise. By considering the ε –insensitive Huber function as a soft insensitive loss function, a novel Bayesian SVR is proposed by Chu et al. (2004). Balasundaram and Meena (2018) introduced asymmetric Huber and asymmetric-insensitive Huber SVR formulations in which the loss functions are expressed as the difference of C1 smooth, squared convex functions. These proposed formulations have the advantage of assuming simplified strongly convex quadratic forms in primal that were solved using a functional iterative approach. By introducing ε −insensitive Huber function in addition to Huber function, robust regression models in primal are constructed by Zhu et al. (2008).

1.3 Organization of the Thesis

There are seven chapters in this thesis. The first chapter is devoted to a review of the literature on support vector machines (SVM). The mathematical formulations and methods of solution for well-known existing SVM variations for classification and regression problems are presented in the second chapter. The next four chapters present our research findings. The last chapter summarizes our research findings and suggests research directions for the future.

Chapter 3

In this chapter, we presented a robust twin bounded support vector machine with pinball loss (Pin-TBSVM) for feature noise affected datasets. Pin-TBSVM is a non-parallel classifier where kernel generated surfaces were determined as the solutions to quadratic programming problems. Experimental results on several benchmark datasets show that the proposed method achieves improved accuracy performance than the popular traditional methods. Though Pin-TBSVM is a simple and efficient learning method, it loses sparsity. An increase in the number of parameters is a concern and the selection of their optimal values is a practical problem which needs attention.

Chapter 4

In this chapter with the aim of obtaining an efficient robust learning model, L_1 −norm-based twin bounded support vector machine with pinball loss for data classification is presented Besides the benefit of less sensitivity to noise property of pinball loss, with the application of L_1 −norm for within-class scatter minimization, the proposed method also enjoys robustness to outliers. As a novel approach to solving, by a simple reformulation of the primal problem considered, an equivalent pair of dual QPPs in m variables only is derived (L1-Pin-TBSVM), where m is the number of training vectors. In comparison with the twin bounded support vector machine (TBSVM), the duals of our proposed L1-Pin-TBSVM are free of inverse matrices and the non-linear duals can be obtained from their linear formulations directly by applying the kernel trick. Experiments on the *Crossplanes* dataset where two or four outliers were introduced show that the proposed L1-Pin-TBSVM outperforms the other SVM methods in terms of accuracy, confirming its robustness to outliers. In addition, numerous empirical results on the two-moon synthetic dataset and several benchmark datasets with varying levels of noise clearly demonstrate improved generalization ability of L1-Pin-TBSVM at comparable training cost which further confirms its effectiveness and suitability where robustness is a problem of major concern. Though L1-Pin-TBSVM is a simple, efficient learning method, it loses sparsity.

Chapter 5

In this chapter, we presented a novel, efficient L_1 −norm-based nonparallel support vector machine classifier with pinball loss (L1-Pin-NPSVM) where its associated optimization problem minimizes the scatter loss and the misclassification error by L_1 −norm and pinball loss respectively with the aim of having the integrated merits of L_1 −norm and pinball loss in achieving enhanced robustness to outliers and bringing noise insensitivity to feature noise. The dual formulation of the proposed method solves a pair of QPPs free of inverse kernel matrices. Our formulation allows a unified framework for the linear and nonlinear kernels. The effectiveness of L1-Pin-NPSVM is evaluated on synthetic and UCI benchmark datasets having outliers and/or being contaminated by noise. The results confirm its superiority in terms of robustness to outliers and noise insensitivity. In summary, we can conclude that the proposed L1-Pin-NPSVM is an efficient method than the other machine-learning methods for real-world problems when robustness and noise insensitivity is of major concern.

Chapter 6

Although the error of misfit measured using the epsilon insensitive function of Vapnik (Vapnik, 2000) leads to a robust regression model, this function is only continuous and therefore the application of popular numerical minimization methods of solving is difficult. In this chapter, the Huber function is used as the error function to measure the data misfit having both the robustness and differentiability properties. Our proposed formulation leads to solving an optimization problem whose solution was obtained by functional iterative methods. Tests with both synthetic and real-world data sets confirm the suitability and applicability of our proposed robust model.

Chapter 7

This final chapter of the thesis summarizes our research findings and identifies future research directions to be pursued.

Chapter 2

Support Vector Machine Techniques

2.1 Introduction

Over the last years, support vector machines (SVMs) proposed by Vapnik et al., in 1990 have emerged as one of the most effective supervised machine learning methods for handling classification and regression problems. SVMs were originally designed to tackle two-class classification problems, but it was later reformulated and expanded to handle multiclass classification problems. (Guyon et al., 2002; Jayadeva et al., 2007; Demsar, 2006; Min & Lee, 2005; Sjoberg et al., 1995; Kumar & Gopal, 2009; Cristianini & Shawe-Taylor). SVMs classify the data samples of two classes by generating a hyper-plane in input space that maximizes the separation between them. They have been widely studied and successfully applied in various pattern recognition areas of research, such as image processing, bioinformatics, and economics (Osuna et al., 1997; Guyon et al., 2002; Min & Lee, 2005). SVMs have gained great attention as a powerful method because of their solid mathematical foundation in statistical learning theory, and good generalization ability to solve nonlinear modelling and classification problems without suffering from many local minima. SVM has numerous advantages in comparison to other machine learning techniques like artificial neural networks (ANN) for example, it provides a global data classification solution. In contrast to other existing data classification algorithms, it generates a unique global hyper-plane to separate data samples of distinct classes rather than local boundaries. Because SVM is based on the Structural Risk Minimization (SRM) concept, it decreases risk during the training phase while also improving generalization. Because of its superior performance, SVM is one of the most widely used data mining classification techniques, with applications in a variety of fields including detection of disease, text categorization, software fault prediction, voice recognition, face recognition, bankruptcy prediction, intrusion detection, time series prediction, music emotion recognition, and so on. (Chen et al., 2011; Ubeyli, 2008; Sweilam et al., 2010; Lee & Kageura, 2007; Wang & Chiang, 2009; Wang & Chiang, 2011; Wang & Chiang, 2007; Elish & Elish, 2008; Can et al., 2013; Ganapathiraju et al., 2004; Chandaka et al., 2009; Manikandan & Venkataramani, 2011; Jonsson et al., 2002; Guo et al., 2001; Shin et al., 2005; Min et al., 2006; Li et al., 2012; Horng et al, 2011; Kuang et al., 2014; Kim, 2003; Han et al., 2009).

In sections 2.2 and 2.3 of this chapter, support vector machines are presented as classification and regression techniques, respectively. SVM with hinge loss and SVM classifier with pinball loss (Pin-SVM) are discussed in subsections 2.2.1 and 2.2.2. In subsections 2.2.3, 2.2.4, and 2.2.5, TWSVM and its regularized variants, twin bounded SVM (TBSVM) and Twin Support Vector machine with pinball loss, are discussed. L1-norm based least-squares TBSVM (L1-LSTBSVM) and Twin Parametric Margin Support Vector Machine (TPMSVM) are discussed in subsections 2.2.6 and 2.2.7. SVR and LS-SVR are discussed in subsections 2.3.1 and 2.3.2. These classification and regression techniques mentioned in this chapter will be utilized as needed in the following chapters to compare our proposed methods.

2.2 Classification Techniques

Suppose we have given a binary classification problem where each and every observation of the classification training data

$$\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m, y_m)\},\ \mathbf{x}_i^t = (x_{i1},\ldots,x_{in})^t \in X \subseteq R^n,\ \ y_i \in Y = \{-1, +1\} \tag{2.1}$$

selected from an unknown probability distribution $p(\mathbf{x}, y)$. Let H stand for a hypothesis set of linear decision functions that maps X to Y, i.e.

$$H = \{\, f \mid f\,(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b\},\ \ \mathbf{w}, \mathbf{x} \in R^n, b \in R \tag{2.2}$$

The loss function is represented by the symbol L: $Y \times Y \rightarrow R^+$ that provides a non-negative real value to any pair $(y_i, \tilde{y}_i)$ where $\tilde{y}_i = f(\boldsymbol{x}_i)$ such as 0-1 loss, linear hinge loss, squared hinge loss, logistic loss, and pinball loss. Table 2.1 contains the definitions of these loss functions, and the Figure 2.1 depicts their graphs. The goal of classification is to discover a hypothesis $f \in H$ that reduces the predicted risk or generalization error given by (Kecman, 2001).

$$R[f] = \underset{\boldsymbol{x}\sim p(\boldsymbol{x},y)}{E}\{L(y, f(\boldsymbol{x}))\} = \int L(y, f(\boldsymbol{x}))p(\boldsymbol{x}, y)d\boldsymbol{x}dy \tag{2.3}$$

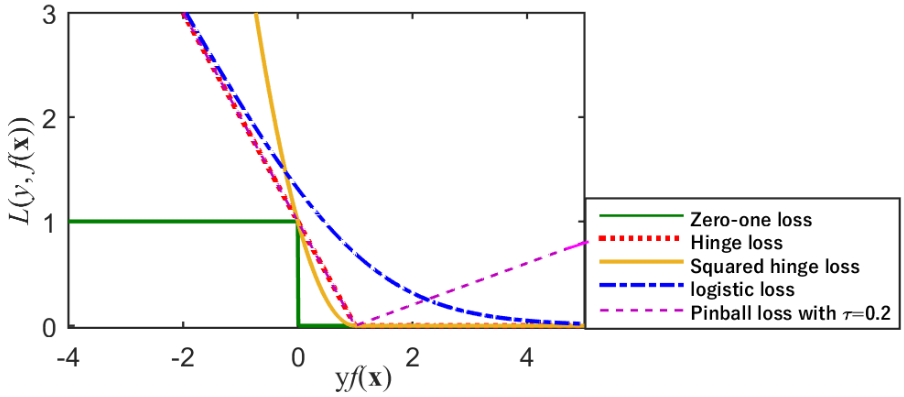


Figure 2.1 Various classification loss functions

Then, as a result of the classification task, a solution will emerge of the form

$$g(\mathbf{x}) = sign(\, f\,(\mathbf{x})) = sign(\mathbf{w}^t\mathbf{x} + b) \tag{2.4}$$

Table 2.1 Various loss functions for classification problem

Loss function	Equations
Zero-one loss	$L_{0/1}(f(\boldsymbol{x}, y) = \begin{cases} 1 & yf(\boldsymbol{x}) \leq 0 \\ 0 & \text{otherwise} \end{cases}$
Hinge loss (Bennett & Mangasaian, 1992)	$L_{hinge}(y, f\,(\mathbf{x})) = \max(0, 1 - yf(\mathbf{x}))$
Squared Hinge loss	$L_{hinge}{}^{2}(y, f\,(\mathbf{x})) = \max(0, 1 - yf(\mathbf{x}))^2$
Logistic loss (Steinwart & Christmann, 2008)	$L_{logistic}(y, f\,(\mathbf{x})) = \ln(1 + \exp(-yf(\mathbf{x})))$
Pinball loss (Huang et al., 2014b)	$L_{\tau}(y, f\,(\mathbf{x})) = \max(yf(\mathbf{x}), -\tau\, yf(\mathbf{x}))$

When dealing with a problem of binary classification, let the vectors of the positive (class +1) and negative classes (class -1) be arranged separately making the rows of matrices $A \in R^{m_1 \times n}$ and $B \in R^{m_2 \times n}$ respectively. Finally, let $C = [A; B] \in R^{m \times n}$ be the matrix of row vectors of the whole training set where $m = m_1 + m_2$ and $\mathbf{x}_i{}^t = (x_{i1},\ldots,x_{in})^t \in R^n$ is a row vector of matrix C that represents i^{th} data point.

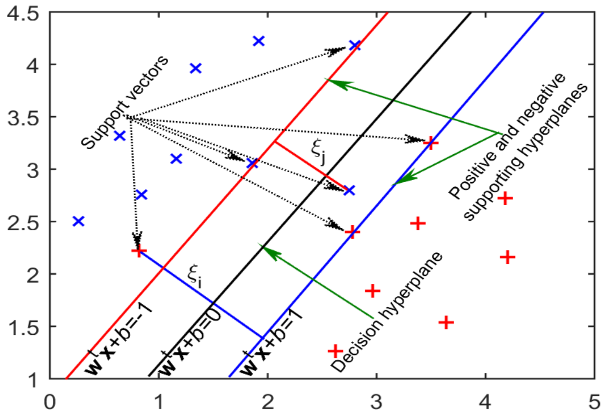


Figure 2.2 Soft margin classifier in two dimensional space

2.2.1 Support Vector Machine with Hinge Loss

In this section, we explain the classical SVM formulation. In the binary classification setting, let the set of training samples be $\{(\boldsymbol{x}_i, y_i)\}_{i=1,2,\ldots,m}$ such that $y_i \in \{-1, +1\}$ be the class label corresponding to the input $\boldsymbol{x}_i \in R^n$. Suppose we have two classes of observations that are linearly separable. Support vector machine aims at determining a separating hyperplane that separates the positive samples from the negative samples with largest margin. The margin of the hyperplane means the shortest distance between the positive and negative samples that are closest to the hyperplane.

we seek a linear SVM classifier to determine a hyperplane such that

$$f(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b = 0 \tag{2.5}$$

so that the margin between two classes is maximized and the training error is minimized. It can be expressed mathematically as a minimization problem with a regularization and empirical risk term as the objective function. By solving the following minimization problem, the unknown normal vector **w** in R^n and scalar threshold b may be determined.

$$\min_{(\boldsymbol{w},b)\in R^{n+1}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + c\sum_{i=1}^{m} L_{hinge}(y_i, f(\boldsymbol{x}_i)) \tag{2.6}$$

Where the empirical risk is measured using the hinge loss and is defined as: $L_{hinge}(\boldsymbol{x}, y, f(\boldsymbol{x})) = max\{\, 0{,}1 - yf(\boldsymbol{x})\}$ and $c > 0$ is a trade-off parameter. With the introduction of the vector of slack variables $\boldsymbol{\xi} = (\xi_1, \dots, \xi_m)^t$, SVM solves the primal problem. So the problem (2.6) can alternatively be stated as a QPP, as shown below.

$$\begin{aligned} &\min_{(\boldsymbol{w},b,\xi)\in R^{n+1+m}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + c\sum_{i=1}^{m} \xi_i \\ &\text{subject to} \quad y_i(\boldsymbol{w}^t\boldsymbol{x}_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0, \quad i = 1, \dots, m \end{aligned} \tag{2.7}$$

Due to measurement or data input errors, real-world training data is not flawless and contains noise. We introduce a slack variable ξ_i for each i^{th} data point as a penalty to capture its deviation from its respective supporting hyperplane so that the supporting hyperplane is unconstrained by it, as shown in Figure 2.2, to reduce the impact of training data points, which are thought to be due to noise, on the size of margin.

Instead of solving the primal optimization problem (2.7), its dual problem is solved by introducing the vector of Lagrangian multipliers $\boldsymbol{\alpha} = [\alpha_1,\dots, \alpha_m]^t$ and $\boldsymbol{\beta} = [\beta_1,\dots, \beta_m]^t$ such that $\alpha_i \geq 0, \beta_i \geq 0$ for all $i = 1,2,\dots,m$, the Lagrangian function for (2.7) can be determined to be

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}||\boldsymbol{w}||^2 + c\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i(y_i(\boldsymbol{w}^t\boldsymbol{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{m}\beta_i\xi_i$$

According to

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{m}\alpha_i y_i \boldsymbol{x}_i = 0, \quad \frac{\partial L}{\partial b} = \sum_{i=1}^{m}\alpha_i y_i = 0, \quad \text{and} \quad \frac{\partial L}{\partial \xi_i} = c - \alpha_i - \beta_i, \forall i = 1, \dots, m$$

the dual problem of (2.7) can be found in the following way:

$$\begin{aligned} &\min_{\alpha\in R^m} \quad \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j\, y_i y_j(\boldsymbol{x}_i^t\boldsymbol{x}_j) - \sum_{i=1}^{m}\alpha_i \\ &\text{subject to} \quad \sum_{i=1}^{m} y_i\alpha_i = 0, \quad 0 \leq \alpha_i \leq c \qquad \text{for } i = 1, \dots, m \end{aligned} \tag{2.8}$$

We obtain the optimal values for dual variables after optimizing the above QPP, and the separating hyperplane (2.5) becomes

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i y_i (\boldsymbol{x}_i^t \cdot \boldsymbol{x}) + b \tag{2.9}$$

The set of support vectors S can be found using the KKT condition, and then b can be calculated as follows: (Abe, 2005)

$$b = \frac{1}{|S|} \sum_{j \in S} \left(y_j - \sum_{i=1}^{m} \alpha_i y_i (\boldsymbol{x}_i^t \cdot \boldsymbol{x}_j) \right) \tag{2.10}$$

The linear SVM decision function as (2.4) is given by

$$g(\boldsymbol{x}) = sign(f(\boldsymbol{x})) = sign\left(\sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i^t \boldsymbol{x} + b \right) \tag{2.11}$$

We've talked about a linear SVM that determines a linear separating hyperplane so far, however many real-world datasets are linearly inseparable, necessitating more expressive non-linear functions than linear separating hyperplanes. Linear SVMs have the unique characteristic of being easily extended to non-linear SVMs (Boser et al., 1992). The strategy is as follows in principle. Through a nonlinear mapping function ϕ: $X \rightarrow F$, all the training data points $\mathbf{x}_i \in R^n$ are transferred from their input space X to a higher dimension dot product space called feature space F, as shown in Figure 2.3. (Boser et al.,1992; Scholkopf & Smola, 2002; Shawe-Taylor & Cristianini, 2004). The SVM then looks for a linear separating hyperplane in F with a maximum margin that corresponds to a nonlinear function in the input space. We can't be sure that the problem will be linearly separable in F, so we'll use slack variables to deal with the misclassified data points once more.

The nonlinear SVM can be simply calculated by solving the dual problem obtained by substituting $\phi(\mathbf{x}_i)^t\mathbf{x}_j$ for the dot product $(\mathbf{x}_i^t \cdot \mathbf{x}_j)$ in the dual problem (2.8) of the linear SVM formulation. In F, the optimal separating hyperplane is given by the following equation (2.12)

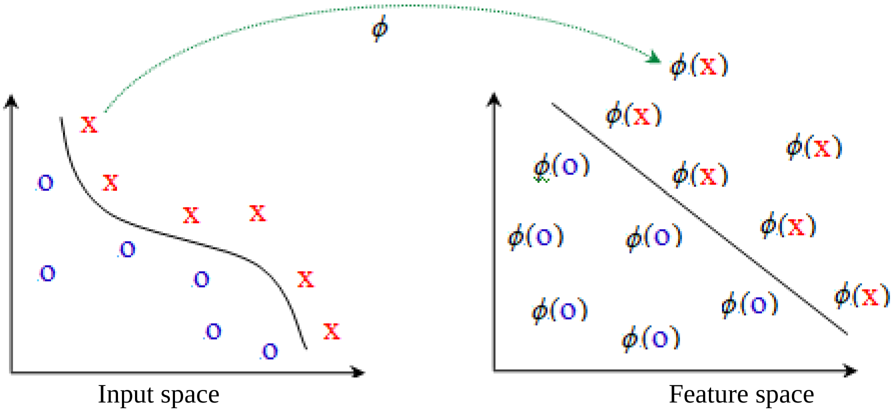


Figure 2.3 Mapping into a higher dimensional feature space (Courtesy: Shawe-Taylor & Cristianini, 2004).

$$f(\boldsymbol{x}) = \boldsymbol{w}^t \phi(\boldsymbol{x}) + b = 0 \tag{2.12}$$

This can be calculated by solving the dual QPP given below

$$\min_{\alpha\in R^m} \quad \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i\alpha_j\, y_i y_j \phi(\boldsymbol{x}_i)^t \phi(\boldsymbol{x}_j) - \sum_{i=1}^{m}\alpha_i$$

$$\text{subject to} \quad \sum_{i=1}^{m} y_i\alpha_i = 0, \quad 0 \le \alpha_i \le c \qquad \text{for } i = 1, \dots, m \tag{2.13}$$

If there is a "kernel function" k such that $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^t\, \mathbf{x}_j$, we can utilize it instead of knowing the function $\phi(\cdot)$ explicitly. The separating hyperplane in F, in this case, can be determined as follows:

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i y_i k(\boldsymbol{x}_i, \boldsymbol{x}) + b \tag{2.14}$$

and

$$b = \frac{1}{|S|}\sum_{j\in S}\left(y_i - \sum_{i=1}^{m} \alpha_i y_i k(\boldsymbol{x}_i, \boldsymbol{x})\right) \tag{2.15}$$

The following formula is used to calculate the nonlinear SVM decision function:

$$g(\boldsymbol{x}) = sign(f(\boldsymbol{x})) = sign\left(\sum_{i=1}^{m} \alpha_i y_i \boldsymbol{k}(\boldsymbol{x}_i, \boldsymbol{x}) + b\right) \tag{2.16}$$

A symmetric function $k(\cdot,\cdot)$ is a kernel function if it meets the Mercer's condition (Cristianini & Shawe-Taylor, 2000), which asserts that for every function $g(\mathbf{x})$, $\int g\,(\mathbf{x})^2\, d\mathbf{x} < \infty$ and the inequality $\int k\,(\mathbf{x},\mathbf{z})\, g(\mathbf{x})\, g(\mathbf{z})\, d\mathbf{x}\, d\mathbf{z} \ge 0$.

The following are some of the most widely utilised kernels:

a.	Linear Kernel :	$k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^t \cdot \mathbf{x}_2$
b.	Polynomial kernel:	$k(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1^t \cdot \mathbf{x}_2 + 1)^d$, $d > 1$
c.	Gaussian/RBF kernel:	$k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\mu\, \lVert\mathbf{x}_1 - \mathbf{x}_2\rVert^2)$, $\mu > 0$
d.	Sigmoidal kernel:	$k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(a\, (\mathbf{x}_1^t \cdot \mathbf{x}_2) + b)$, $a > 0, b > 0$

where $\mathbf{x}_1, \mathbf{x}_2 \in R^n$ and d, μ, a, b are user defined kernel parameters.

2.2.2 Support Vector Machine with Pinball Loss

Though SVM provides a sparse global solution, however, since it uses hinge loss to maximize the distance between the closest data points of the two classes, its resulting classifier is prone to feature noise and is unstable for re-sampling. As a significant approach to overcome this drawback, a robust SVM based on pinball loss (Pin-SVM) is proposed in (Haung et al., 2014b) where for measuring the training error the pinball loss is considered and it is defined as (Haung et al., 2014b)

$$L_\tau(\boldsymbol{x}, y, f(\boldsymbol{x})) = \begin{cases} 1 - yf(\boldsymbol{x}), & 1 - yf(\boldsymbol{x}) \ge 0, \\ -\tau(1 - yf(\boldsymbol{x})), & 1 - yf(\boldsymbol{x}) < 0, \end{cases} \tag{2.17}$$

where $0 \leq \tau \leq 1$ is a parameter. Pinball loss is illustrated as a geometrical representation for a variety of values of τ is shown in Figure 2.4.

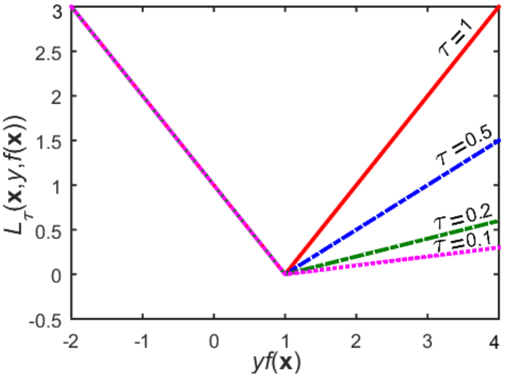


Figure 2.4 Plots of pinball loss

Note that $L_\tau(\boldsymbol{x}, y, f(\boldsymbol{x})) = max\{0, (1 - yf(\boldsymbol{x}))\} + \tau\, max\{0, -(1 - yf(\boldsymbol{x}))\}$. Clearly, when $\tau = 0$ and $\tau = 1$ imply the pinball loss becomes the hinge loss and absolute loss respectively. Thus, pinball can be regarded as a trade-off considering the hinge loss and L_1 −norm loss and it combines the advantages of both of them.

Following the problem formulation of hinge SVM, the linear pinball loss support vector machine (Pin-SVM) (Haung et al., 2014b) can be obtained in the form

$$\begin{aligned} &\min_{(\boldsymbol{w},b,\boldsymbol{\xi})\in R^{n+1+m}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + c\sum_{i=1}^{m}\xi_i \\ &\text{subject to} \qquad y_i(\boldsymbol{w}^t\boldsymbol{x}_i + b) \geq 1 - \xi_i, \\ &\qquad\qquad y_i(\boldsymbol{w}^t\boldsymbol{x}_i + b) \leq 1 + \frac{1}{\tau}\xi_i, \quad i = 1, \dots, m. \end{aligned} \tag{2.18}$$

By introducing Lagrange vectors $\boldsymbol{\alpha} = (\alpha_1,\dots,\alpha_m)^t$ and $\boldsymbol{\beta} = (\beta_1,\dots, \beta_m)^t$ the dual of (2.18) can be obtained as follows

$$\begin{aligned} &\min_{(\boldsymbol{\alpha},\boldsymbol{\beta})\in R^{m+m}} \quad \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j\, (\mathbf{x}_i^t \cdot \mathbf{x}_j)(\alpha_i - \beta_i)(\alpha_i - \beta_i) - \sum_{i=1}^{m}(\alpha_i - \beta_i) \\ &\text{subject to} \quad \sum_{i=1}^{m}(\alpha_i - \beta_i)y_i = 0, \\ &\qquad \alpha_i + \frac{1}{\tau}\beta_i = c, \alpha_i, \beta_i \geq 0 \text{ for } i = 1, \dots m \end{aligned} \tag{2.19}$$

We get the optimal values for Lagrangian vectors after optimizing (2.19), and then **w** and b can be calculated. For the linear situation, the Pin-SVM decision function is

$$g(\mathbf{x}) = sign(\sum_{i=1}^{m}(\alpha_i - \beta_i)(\mathbf{x}_i^t \cdot \mathbf{x}_i) + b) \tag{2.20}$$

The dual problem for the nonlinear case can be obtained as (2.20), using the kernel technique.

$$\min_{(\boldsymbol{\alpha},\boldsymbol{\beta})\in R^{m+m}} \quad \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j\, k(\mathbf{x}_i^t\cdot\mathbf{x}_j)(\alpha_i - \beta_i)(\alpha_i - \beta_i) - \sum_{i=1}^{m}(\alpha_i - \beta_i) \tag{2.21}$$

$$\text{subject to} \quad \sum_{i=1}^{m}(\alpha_i - \beta_i)y_i = 0,$$

$$\alpha_i + \frac{1}{\tau}\beta_i = c, \alpha_i, \beta_i \geq 0 \text{ for } i = 1, \dots m$$

where $k(\cdot,\cdot)$ is a kernel function of your choice. The decision function for nonlinear Pin-SVM can be obtained after the solution of (2.21) is known. Pin-SVM's computational complexity is comparable to that of the SVM with hinge loss. It's easy to see how pinball loss penalizes all successfully recognized patterns, bringing them closer to the classifier. The major advantage of penalizing the correctly classified patterns is that with the increase in the value of τ, the width of the margin increases and thereby insensitivity with respect to noise around the decision boundary will be achieved. For a more in-depth analysis on Pin-SVM, see (Haung et al., 2014b).

2.2.3 Twin Support Vector Machine

With the aim of reducing training costs, following the work on nonparallel SVM called Proximal SVM via Generalized Eigenvalues (GEPSVM) introduced in (Mangasarian & Wild, 2006), twin SVM (TWSVM) for binary classification was proposed recently by Jayadeva et al., (2007). TWSVM seeks two nonparallel hyperplanes with the property that each one of them is as close as possible to inputs of one of the two classes and at the same time at least one unit distance away from the inputs of the other class. Despite its excellent generalization performance, learning by SVM is much more expensive when the size of the problem is large. However, since it solves two smaller sized QPPs leading to approximately four times faster learning speed than SVM, the TWSVM (Jayadeva et al., 2007). becomes a very popular method in the literature of machine learning.

Let $T = \{(\boldsymbol{x}_1, +1), \dots, (\boldsymbol{x}_{m_1}, +1), (\boldsymbol{x}_{m_1+1}, -1), \dots, (\boldsymbol{x}_{m_1+m_2}, -1)\}$ be the set of training vectors considered. Let the vectors of the positive and negative classes be arranged separately making the rows of matrices $A \in R^{m_1 \times n}$ and $B \in R^{m_2 \times n}$ respectively. Finally, let $C = [A; B]$ be the matrix of row vectors of the whole training set.

The linear TWSVM seeks a pair of non-parallel hyperplanes of the form

$$f_1(\mathbf{x}) = \mathbf{w_1^t}\mathbf{x} + b_1 = 0 \text{ and } f_2(\mathbf{x}) = \mathbf{w_2^t}\mathbf{x} + b_2 = 0 \tag{2.22}$$

TBSVM solves the pair of QPPs of the form below

$$\min_{(\boldsymbol{w}_1, b_1, \xi_2)\in R^{n+1+m_2}} \quad \frac{1}{2}||A\mathbf{w}_1 + \mathbf{e}_1 b_1||^2 + c_1\mathbf{e}_2^t\xi_2$$

$$\text{subject to} \quad -(B\mathbf{w}_1 + \mathbf{e}_2 b_1) + \xi_2 \geq \mathbf{e}_2, \ \xi_2 \geq \mathbf{0} \tag{2.23a}$$

and

$$\min_{(w_2,b_2,\xi_1)\in R^{n+1+m_1}} \frac{1}{2}||B\mathbf{w}_2 + \mathbf{e}_2 b_2||^2 + c_2\mathbf{e}_1^t\xi_1$$
$$\text{subject to} \quad (A\mathbf{w}_2 + \mathbf{e}_1 b_2) + \xi_1 \geq \mathbf{e}_1,\ \ \xi_1 \geq \mathbf{0} \tag{2.23b}$$

where for $k = 1,2, c_k, c_{k+1} > 0$ are regularization parameters, $\boldsymbol{\xi}_k \in R^{m_k}$ is a vector of slack variables, $b_k \in R$, are unknowns and $\boldsymbol{e}_k \in R^{m_k}$ is the vector of ones. By solving their Lagrangian duals and applying KKT conditions, the solution of the pair of QPPs (2.23a) and (2.23b) can be derived indirectly. The dual of QPPs (2.23a) and (2.24b) can be obtained using the Lagrangian vectors $\mathbf{u}_1 \in R^{m_2}$ and $\mathbf{u}_2 \in R^{m_1}$.

$$\min_{\boldsymbol{u}_1\in R^{m_2}} \frac{1}{2}\mathbf{u}_1^t H(G^tG)^{-1}H^t\mathbf{u}_1 - \mathbf{e}_2^t\mathbf{u}_1$$
$$\text{subject to} \quad \mathbf{0} \leq \mathbf{u}_1 \leq c_1\mathbf{e}_2 \tag{2.24a}$$

and

$$\min_{\boldsymbol{u}_2\in R^{m_1}} \frac{1}{2}\mathbf{u}_2^t G(H^tH)^{-1}G^t\mathbf{u}_2 - \mathbf{e}_1^t\mathbf{u}_2$$
$$\text{subject to} \quad \mathbf{0} \leq \mathbf{u}_2 \leq c_2\mathbf{e}_1 \tag{2.24b}$$

$G = [A\ \mathbf{e}_1]$, $H = [B\ \mathbf{e}_2]$ are augmented matrices, respectively. The augmented vectors $[\mathbf{w}_1^t\ b_1]^t$ and $[\mathbf{w}_2^t\ b_2]^t$ can be generated as

$$[\mathbf{w}_1^t\ b_1] = -\,(G^tG)^{-1}H^t\mathbf{u}_1 \text{ and } [\mathbf{w}_2^t\ b_2] = (H^tH)^{-1}G^t\mathbf{u}_2.$$

The augmented vectors are generated after optimizing the dual QPPs (2.24a) and (2.24b), and then any new data point $\mathbf{x} \in R^n$ is allocated to class r_k ($r_1 = +1$, $r_2 = -1$), based on its proximity to the non-parallel hyperplanes, i.e.

$$k = \arg\min_{k=1,2} |\,\mathbf{x}^t\mathbf{w}^k + b^{\mathrm{k}}\,| \tag{2.25}$$

where the perpendicular distance between the data point $\mathbf{x}$ and the hyperplane $\mathbf{x}^t\,\mathbf{w}_k + b_{\mathrm{k}}$ is given by $|\,\mathbf{x}^t\,\mathbf{w}_k + b_{\mathrm{k}}\,|$.

The nonlinear TWSVM looks for a pair of kernel generated surfaces to handle the linearly inseparable data of the form

$$K(\mathbf{x}^t, C^t)\,\mathbf{w}_1 + b_1 = 0 \text{ and } K(\mathbf{x}^t, C^t)\,\mathbf{w}_2 + b_2 = 0 \tag{2.26}$$

$K(\mathbf{x}^t, C^t) = (k(\mathbf{x}, \mathbf{x}_1),\ldots,k(\mathbf{x}, \mathbf{x}_m))$ is a row vector in R^m, and $C = [A; B] \in R^{(m_1+m_2)\times n}$. Solving the following QPPs obtains the kernel generated surfaces (2.26)

$$\min_{(w_1,b_1,\xi_2)\in R^{m+1+m_1}} \frac{1}{2}||\,K(A, C^t)\mathbf{w}_1 + \mathbf{e}_1 b_1||^2 + c_1\mathbf{e}_2^t\xi_2$$
$$\text{subject to} \quad -\,(K(B, C^t)\,\mathbf{w}_1 + \mathbf{e}_2 b_1) + \xi_2 \geq \mathbf{e}_2,\ \ \xi_2 \geq \mathbf{0} \tag{2.27a}$$

and

$$\min_{(w_2,b_2,\xi_1)\in R^{m+1+m_1}} \frac{1}{2}||\,K(B, C^t)\mathbf{w}_2 + \mathbf{e}_2 b_2||^2 + c_2\mathbf{e}_1^t\xi_1$$
$$\text{subject to} \quad (K(A, C^t)\mathbf{w}_2 + \mathbf{e}_1 b_2) + \xi_1 \geq \mathbf{e}_1,\ \ \xi_1 \geq \mathbf{0} \tag{2.27b}$$

respectively, where $\mathbf{w}_1, \mathbf{w}_2 \in R^m$ and $b_1, b_2 \in R$.

The duals of the given QPPs can be constructed using the Lagrangian vectors $\mathbf{u}_1 \in R^{m_2}$ and $\mathbf{u}_2 \in R^{m_1}$, respectively.

$$\min_{\boldsymbol{u}_1 \in R^{m_2}} \quad \frac{1}{2}\, \mathbf{u}_1^t H(G^t G)^{-1} H^t \mathbf{u}_1 - \mathbf{e}_2^t \mathbf{u}_1$$

$$\text{subject to} \qquad \mathbf{0} \leq \mathbf{u}_1 \leq c_1 \mathbf{e}_2 \tag{2.28a}$$

and

$$\min_{\boldsymbol{u}_2 \in R^{m_1}} \quad \frac{1}{2}\, \mathbf{u}_2^t G(H^t H)^{-1} G^t \mathbf{u}_2 - \mathbf{e}_1^t \mathbf{u}_2$$

$$\text{subject to} \qquad \mathbf{0} \leq \mathbf{u}_2 \leq c_2 \mathbf{e}_1 \tag{2.28b}$$

In the above given equations $G = [K(A, C^t) \;\; \mathbf{e}_1]$, $H = [K(B, C^t) \;\; \mathbf{e}_2]$. The augmented vectors $[\mathbf{w}_1^t \; b_1]^t$ and $[\mathbf{w}_2^t \; b_2]^t$ can be generated as

$$[\mathbf{w}_1^t \; b_1] = -\, (G^t G)^{-1} H^t \mathbf{u}_1 \text{ and } [\mathbf{w}_2^t \; b_2] = (H^t H)^{-1} G^t \mathbf{u}_2.$$

After optimizing the above dual QPPs to obtain these augmented vectors, any new data point $\mathbf{x} \in R^n$ is allocated to class r_k ($r_1 = +1$, $r_2 = -1$), based on its proximity to the kernel generated surfaces, i.e.

$$k \; = \; \arg \min_{k=1,2} |\, K(\mathbf{x}^t, C^t)\mathbf{w}^k \; + \; b^{\mathrm{k}} \;| \tag{2.29}$$

2.2.4 Twin Bounded Support Vector Machine

Implementing the structural risk minimization principle as the backbone for obtaining better performance of the classifier, twin bounded SVM (TBSVM) is proposed in (Shao et al., 2011) as an extension of TWSVM. The structural risk minimization concept is implemented by introducing the regularization term, which is a substantial advantage of TBSVM over TWSVM. The following primal TBSVM formulation for linear case is obtained by adding the regularisation terms in TWSVM's primal QPPs (2.23a) and (2.23b)

$$\min_{(\boldsymbol{w}_1, b_1, \boldsymbol{\xi}_2) \in R^{n+1+m_2}} \quad \frac{1}{2} c_3 (||\mathbf{w}_1||^2 + b_1^2) + \frac{1}{2} ||A\mathbf{w}_1 + \mathbf{e}_1 b_1||^2 + c_1 \mathbf{e}_2^t \boldsymbol{\xi}_2$$

$$\text{subject to } -(B\mathbf{w}_1 + \mathbf{e}_2 b_1) + \boldsymbol{\xi}_2 \geq \mathbf{e}_2, \;\; \boldsymbol{\xi}_2 \geq \mathbf{0} \tag{2.30a}$$

and

$$\min_{(\boldsymbol{w}_2, b_2, \boldsymbol{\xi}_1) \in R^{n+1+m_1}} \quad \frac{1}{2} c_4 (||\mathbf{w}_2||^2 + b_2^2) + \frac{1}{2} ||B\mathbf{w}_2 + \mathbf{e}_2 b_2||^2 + c_2 \mathbf{e}_1^t \boldsymbol{\xi}_1$$

$$\text{subject to} \qquad (A\mathbf{w}_2 + \mathbf{e}_1 b_2) + \boldsymbol{\xi}_1 \geq \mathbf{e}_1, \;\; \boldsymbol{\xi}_1 \geq \mathbf{0} \tag{2.30b}$$

The dual of the abovementioned pair of QPPs (2.30a) and (2.30b) can be constructed using the Lagrangian vectors $\mathbf{u}_1 \in R^{m_2}$ and $\mathbf{u}_2 \in R^{m_1}$.

$$\min_{\boldsymbol{u}_1 \in R^{m_2}} \quad \frac{1}{2}\, \mathbf{u}_1^t H(G^t G + c_3 I)^{-1} H^t \mathbf{u}_1 - \mathbf{e}_2^t \mathbf{u}_1$$

$$\text{subject to} \qquad \mathbf{0} \leq \mathbf{u}_1 \leq c_1 \mathbf{e}_2 \tag{2.31a}$$

and

$$\min_{\boldsymbol{u}_2 \in R^{m_1}} \quad \frac{1}{2}\mathbf{u}_2^t G(H^tH + c_4I)^{-1}G^t\mathbf{u}_2 - \mathbf{e}_1^t\mathbf{u}_2$$

subject to $\quad \mathbf{0} \leq \mathbf{u}_2 \leq c_2\mathbf{e}_1$ (2.31b)

where $G = [A \;\; \mathbf{e}_1]$, $H = [B \;\; \mathbf{e}_2]$. The augmented vectors $[\mathbf{w}_1^t \; b_1]^t$ and $[\mathbf{w}_2^t \; b_2]^t$ can be obtained as

$$[\mathbf{w}_1^t \; b_1] = -(G^tG + c_3)^{-1}H^t\mathbf{u}_1 \text{ and } [\mathbf{w}_2^t \; b_2] = (H^tH + c_4)^{-1}G^t\mathbf{u}_2$$

In a similar way, The kernel-produced surfaces (2.26) for TBSVM can be calculated using the following primal TBSVM formulation for nonlinear cases by adding the regularisation terms in primal QPPs (2.27a) and (2.27b) of TWSVM.

$$\min_{(\boldsymbol{w}_1, b_1, \xi_2) \in R^{m+1+m_1}} \quad \frac{1}{2}c_3(\|\mathbf{w}_1\|^2 + b_1^2) + \frac{1}{2}\| K(A, C^t)\mathbf{w}_1 + \mathbf{e}_1b_1\|^2 + c_1\mathbf{e}_2^t\xi_2$$

subject to $-(K(B, C^t)\,\mathbf{w}_1 + \mathbf{e}_2b_1) + \xi_2 \geq \mathbf{e}_2, \;\; \xi_2 \geq \mathbf{0}$ (2.32a)

and

$$\min_{(\boldsymbol{w}_2, b_2, \xi_1) \in R^{m+1+m_1}} \quad \frac{1}{2}c_4(\|\mathbf{w}_2\|^2 + b_2^2) + \frac{1}{2}\| K(B, C^t)\mathbf{w}_2 + \mathbf{e}_2b_2\|^2 + c_2\mathbf{e}_1^t\xi_1$$

subject to $\quad (K(A, C^t)\mathbf{w}_2 + \mathbf{e}_1b_2) + \xi_1 \geq \mathbf{e}_1, \;\; \xi_1 \geq \mathbf{0}$ (2.32b)

Asssuming $G = [K(A, C^t) \;\; \mathbf{e}_1]$, and $H = [K(B, C^t) \;\; \mathbf{e}_2]$ in (2.32), the duals of (2.32a) and (2.32b) are equivalent to (2.31a) and (2.31b) respectively.

Any new data point $\mathbf{x} \in R^n$ is allocated to class r_k ($r_1 = +1$, $r_2 = -1$) by (2.25) for linear case and (2.29) for nonlinear case once the augmented vectors $[\mathbf{w}_1^t \; b_1]^t$ and $[\mathbf{w}_2^t \; b_2]^t$ of TBSVM are known by optimising its dual QPPs. The reader is directed to TBSVM for more information (Shao et. al., 2011)

2.2.5 Twin Support Vector Machine with Pinball Loss

Peng (2011a) suggested a twin parametric-margin SVM model (TPMSVM) based on TWSVM. wherein two nonparallel hyperplanes are obtained by solving two smaller sized QPPs similar to TWSVM. This model is suitable for the heteroscedastic noise that strongly depends on the input values. However, it employs the noise sensitive hinge loss function (Xu et.al, 2016). To further improve the generalization performance, by employing pinball loss in TPMSVM instead of hinge loss, Xu et al. (2016) proposed a novel Pin-TSVM model for datasets corrupted by noise. In this section, we briefly discuss Pin-TSVM formulation proposed in (Xu et.al, 2016).

Assume that matrix $C \in R^{m \times n}$ represents training set of class +1 and class -1, i.e. containing m_1 number of data points of class +1 represented by a matrix $A = [A_1;\ldots;A_{m1}] \in R^{m1 \times n}$ and m_2 number of data points of class -1 represented by a matrix $B \in R^{m2 \times n}$ so that $C = [A; B]$ and $m = m_1 + m_2$.

Supposed that the training data is mapped in a feature space through a nonlinear mapping $\phi(\cdot)$. Then, in the feature space nonlinear Pin-TSVM detects two nonparallel hyperplanes

$$f_1(\mathbf{x}) = \mathbf{w}_1{}^t\phi(\mathbf{x}) + b_1 = 0 \text{ and } f_2(\mathbf{x}) = \mathbf{w}_2{}^t\phi(\mathbf{x}) + b_2 = 0 \tag{2.33}$$

which can be determined by solving the following QPPs

$$\begin{aligned} \min_{w_1, b_1, \xi_1} \quad & \frac{1}{2}\|\mathbf{w}_1\|^2 + \frac{v_1}{m_2}(\,\phi(B)\,\mathbf{w}_1 + b_1\mathbf{e}_2) + \frac{c_1}{m_1}\mathbf{e}_1{}^t\,\xi_1 \\ \text{subject to} \quad & \phi(A)\,\mathbf{w}_1 + b_1\mathbf{e}_1 \geq -\xi_1 \\ & \phi(A)\,\mathbf{w}_1 + b_1\mathbf{e}_1 \leq \frac{1}{\tau_1}\xi_1 \end{aligned} \tag{2.34a}$$

and

$$\begin{aligned} \min_{w_2, b_2, \xi_2} \quad & \frac{1}{2}\|\mathbf{w}_2\|^2 - \frac{v_2}{m_1}(\,\phi(A)\,\mathbf{w}_2 + b_2\mathbf{e}_1) + \frac{c_2}{m_2}\mathbf{e}_2{}^t\,\xi_2 \\ \text{subject to} \quad & -(\phi(B)\,\mathbf{w}_2 + b_2\mathbf{e}_2) \geq -\xi_2 \\ & -(\phi(B)\,\mathbf{w}_2 + b_2\mathbf{e}_2) \leq \frac{1}{\tau_2}\,\xi_2 \end{aligned} \tag{2.34b}$$

where $v_i > 0$, $c_i > 0 : i = 1, 2$ are user defined parameters; $\xi_1 \in R^{m_1}$, $\xi_2 \in R^{m_2}$ are slack vectors; $\phi(A) = [\phi(A_{1,})\ ;\ldots;\ \phi(A_{m_1})]$ and $\phi(B) = [\phi(B_{1,})\ ;\ \ldots\ ;\ \phi(B_{m_2})]$. With the Lagrangian vectors $\boldsymbol{\alpha}_1$, $\boldsymbol{\beta}_1 \in R^{m_1}$ and $\boldsymbol{\alpha}_2$, $\boldsymbol{\beta}_2 \in R^{m_2}$, the duals of (2.34a) and (2.34b) can be obtained, respectively, as

$$\begin{aligned} \min_{(\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1) \in R^{m_1 + m_1}} \quad & \frac{1}{2}(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1)^t K(A, A^t)\,(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1) - \frac{v_1}{m_2}\mathbf{e}_2{}^t K(B, A^t)\,(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1) \\ \text{subject to} \quad & \mathbf{e}_1{}^t\,(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1) = v_1 \\ & \boldsymbol{\alpha}_1 + \frac{1}{\tau_1}\boldsymbol{\beta}_1 = \frac{c_1}{m_1}\mathbf{e}_1, \quad \boldsymbol{\alpha}_1 \geq \mathbf{0},\ \boldsymbol{\beta}_1 \geq \mathbf{0} \end{aligned} \tag{2.35a}$$

and

$$\begin{aligned} \min_{(\boldsymbol{\alpha}_2, \boldsymbol{\beta}_2) \in R^{m_2 + m_2}} \quad & \frac{1}{2}(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2)^t K(B, B^t)\,(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2) - \frac{v_2}{m_1}\mathbf{e}_1{}^t K(A, B^t)(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2) \\ \text{subject to} \quad & \mathbf{e}_2{}^t\,(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2) = v_2 \\ & \boldsymbol{\alpha}_2 + \frac{1}{\tau_2}\boldsymbol{\beta}_2 = \frac{c_2}{m_2}\mathbf{e}_2, \quad \boldsymbol{\alpha}_2 \geq \mathbf{0},\ \boldsymbol{\beta}_2 \geq \mathbf{0} \end{aligned} \tag{2.35b}$$

where $K(A, B^t) \in R^{m1 \times m2}$; $K(A, B^t) = (k(A_i, B_1), \ldots, k(A_i, Bm_2))$ is the i^{th} row vector of $K(A, B^t)$; $k(A_i, B_j) = \phi(A_i)\ \phi(B_j)^t$ and $k(\cdot)$ is an appropriately chosen kernel function. For a detailed discussion on the problem formulation of Pin-TSVM, its method of solution and advantages, see (Xu et al, 2016).

2.2.6 L1-norm based least squares TBSVM (L1-LSTBSVM)

Although TBSVM achieves improved classification performance by implementing SRM principle and is faster than TWSVM, one needs to solve two QPPs (2.32a)-(2.32b) which is still computationally expensive. To address this problem, least-squares TBSVM based on L_1 −norm (L1-LSTBSVM) has been proposed recently (Yan et al., 2018) where the inequality constraints of TBSVM are replaced by equalities and thereby system of linear equations is solved instead of QPPs. In addition to it, as an alternative to L_2 −norm, L_1 −norm is used for intra-class compactness so that enhanced robustness to outliers is achieved.

L1-LSTBSVM determines two non-parallel hyperplanes of the form

$$f_1(\mathbf{x}) = \boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0 \text{ and } f_2(\mathbf{x}) = \boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0 \tag{2.36}$$

by solving the following pair of optimization problems (Yan et al., 2018)

$$\begin{aligned} &\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2} \quad \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1||_1 + \frac{1}{2}c_1\boldsymbol{\xi}_2^t\boldsymbol{\xi}_2 \\ &\text{subject to} \qquad -(B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) + \boldsymbol{\xi}_2 = \boldsymbol{e}_2, \end{aligned} \tag{2.37a}$$

and

$$\begin{aligned} &\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2||_1 + \frac{1}{2}c_2\boldsymbol{\eta}_1^t\boldsymbol{\eta}_1 \\ &\text{subject to} \qquad (A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1) + \boldsymbol{\eta}_1 = \boldsymbol{e}_1, \end{aligned} \tag{2.37b}$$

where $c_k > 0$ and $c_{k+1} > 0$ are user defined parameters; $\boldsymbol{\xi}_k, \boldsymbol{\eta}_k \in R^{m_k}$ are slack vectors; $\boldsymbol{e}_k \in R^{m_k}$ is the vector of ones and k = 1, 2. It is proposed to solve the above problems by an iterative algorithm (Yan et al., 2018). Though L1-LSTBSVM shows satisfactory performance in terms of accuracy and learning speed, its extension to non-linear kernel needs further study (Yan et al., 2018). It is worth noting that the least-squares (LS) loss applied for misclassification is not robust to outliers and consequently we can say that L1-LSTBSVM is not high in robustness.

2.2.7 Twin Parametric Margin Support Vector Machine (TPMSVM)

For the purpose of having the combined advantages of faster learning speed and flexible parametric-margin, motivated by the study of (Jayadeva et al, 2007) and parametric-margin ν −support vector machine (par-ν −SVM) (Hao, 2010), a novel twin parametric-margin support vector machine (TPMSVM) for classification was proposed in (Peng, 2011a). The non-parallel parametric-margin hyperplanes of TPMSVM are derived by solving a pair of QPPs of smaller size. With the non-linear mapping $\phi(.)$into a feature space, the non-parallel hyperplanes defined by

$$f_1(\boldsymbol{x}) = \boldsymbol{w}_1^t\phi(\boldsymbol{x}) + b_1 = 0 \text{ and } f_2(\boldsymbol{x}) = \boldsymbol{w}_2^t\phi(\boldsymbol{x}) + b_2 = 0 \tag{2.38}$$

are obtained as the solutions of the following pair of optimization problems

$$\begin{aligned} &\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\eta}_1} \quad \frac{1}{2}||\boldsymbol{w}_1||^2 + \frac{\nu_1}{m_2}\boldsymbol{e}_2^t(\phi(B)\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) + \frac{c_1}{m_1}\boldsymbol{e}_1^t\boldsymbol{\eta}_1 \\ &\text{subject to} \qquad \phi(A)\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 \geq \boldsymbol{0}_1 - \boldsymbol{\eta}_1, \\ &\qquad\qquad\qquad \boldsymbol{\eta}_1 \geq \boldsymbol{0}_1 \end{aligned} \tag{2.39a}$$

and

$$\begin{aligned} &\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_2} \quad \frac{1}{2}||\boldsymbol{w}_2||^2 - \frac{\nu_2}{m_1}\boldsymbol{e}_1^t(\phi(A)\boldsymbol{w}_2 + b_2\boldsymbol{e}_1) + \frac{c_2}{m_2}\boldsymbol{e}_2^t\boldsymbol{\eta}_2 \\ &\text{subject to} \qquad -(\phi(B)\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \geq -\boldsymbol{\eta}_2, \end{aligned} \tag{2.39b}$$

$$\boldsymbol{\eta}_2 \geq \mathbf{0}_2$$

where $v_k > 0, c_k > 0 (k = 1,2)$ are user defined parameters; $\boldsymbol{\eta_1} \in R^{m_1}, \boldsymbol{\eta_2} \in R^{m_2}$are vectors of slack variables and $\phi(A)$=[$\phi(A_{1,})$;…; $\phi(Am_1)$] and $\phi(B)$=[$\phi(B_{1,})$;…; $\phi(Bm_2)$] are matrices of row vectors $\phi(A_\mathrm{i})$ and $\phi(B_\mathrm{j})$. With the first term of the objective function, the structural risk is minimized. Minimization of the second term leads to negative (positive) class data points to be far away from the positive (negative) hyperplane (Peng, 2011a; Xu et al, 2016). Whereas in the third term, the positive (negative) class data points lying on the wrong side of the positive (negative) hyperplane are taken as misclassified data points and the sum of the misclassification error is minimized. TPMSVM solves for parametric-margin hyperplanes with flexible margins and thus it is suitable for data having heteroscedastic error structure. However, it loses sparsity. For more details on TPMSVM, see (Peng, 2011a).

2.3 Regression Techniques

Suppose we have given a regression problem where each and every observation of the regression training data

$$\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m, y_m)\},\ \mathbf{x}_i^t = (x_{i1},\ldots,x_{in})^t \in X \subseteq R^n,\ \ y_i \in Y \subseteq R \tag{2.40}$$

selected from an unknown probability distribution $p(\mathbf{x}, y)$. Let H stand for a hypothesis set of linear decision functions that maps X to Y, i.e.

$$H = \{ f \mid f(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b\},\ \ \mathbf{w}, \mathbf{x} \in R^n, b \in R \tag{2.41}$$

The loss function is represented by the symbol L: $Y \times Y \rightarrow R^+$ that provides a non-negative real value to any pair $(y_i, \tilde{y}_i)$ where $\tilde{y}_i = f(\boldsymbol{x}_i)$ such as Laplacian loss, Gaussian loss, ε-insensitive loss, quadratic ε-insensitive loss and Huber loss. Table 2.2 contains the definitions of these loss functions, and Figure 2.5 depicts their graphs.

Table 2.2 Various loss functions for regression problem

Loss function	Equations
Laplacian or Linear loss	$L_1(y, f(\mathbf{x})) = \mid y - f(\mathbf{x}) \mid$
Gaussian or Quadratic loss	$L_2(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$
(Linear) ε-insensitive loss (Vapnik, 2000)	$L^{\varepsilon}(y, f(\mathbf{x})) = \max(0, \mid y - f(\mathbf{x}) \mid - \varepsilon)$
Quadratic ε-insensitive loss (Gunn, 1998)	$L_2^{\varepsilon}(y, f(\mathbf{x})) = \max(0, \mid y - f(\mathbf{x}) \mid - \varepsilon)^2$
Huber loss (Bernhard & Scholkopf, 2002)	$L_H(y, f(\boldsymbol{x})) = \begin{cases} (y - f(\boldsymbol{x}))^2, & \text{if } \lvert y - f(\boldsymbol{x})\rvert \leq \gamma \\ 2\gamma\lvert y - f(\boldsymbol{x})\rvert - \gamma^2 & \text{otherwise} \end{cases}$

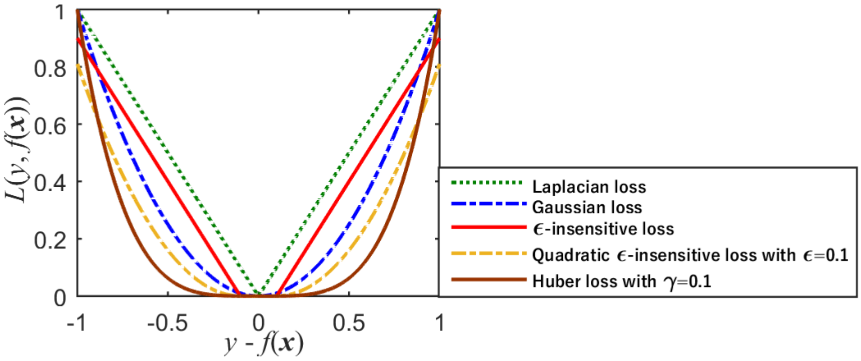


Figure 2.5 Various loss functions for regression

The goal of regression is to discover a hypothesis $f \in H$ that reduces the predicted risk or generalization error given by (Kecman, 2001).

$$R[f] = \underset{\boldsymbol{x} \sim p(\boldsymbol{x},y)}{E}\{L(y, f(\boldsymbol{x}))\} = \int L(y, f(\boldsymbol{x}))p(\boldsymbol{x}, y)d\boldsymbol{x}dy \qquad (2.42)$$

2.3.1 Support Vector Regression

The support vector machine has been used to solve regression problems, such as function approximation. It is a regression-oriented version of SVM that retains all of the key aspects of the maximal margin technique, such as duality, sparseness, kernel, and convexity. Vapnik proposed the Support Vector Regression (SVR) algorithm, which uses an ε-insensitive loss function (2000).

The linear SVR searches for a function $f(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b$ that minimizes the training error while also being as flat as possible, i.e. $||\mathbf{w}||$ is made as small as possible, in order to discover an optimal regressor (approximation function) for the regression problem specified in section 2.3. In terms of mathematics, this entails

$$\min_{(\boldsymbol{w},b)\in R^{n+1}} \frac{1}{2}||\mathbf{w}||^2 + c\sum_{i=1}^{m} L^{\varepsilon}\,(y_i, f(\,\mathbf{x}_i, y_i)) \qquad (2.43)$$

$L^{\varepsilon}(y, f(\mathbf{x})) = \max(0, |y - f(\mathbf{x})| - \varepsilon)$ is is the ε-insensitive loss function, where $c > 0$ is a regularisation parameter. The problem (2.43) can also be expressed as follows:

$$\min_{\boldsymbol{w},b,\xi_1,\xi_2} \frac{1}{2}||\mathbf{w}||^2 + c\sum_{i=1}^{m}(\xi_{1i} + \xi_{2i}) \qquad (2.44)$$

subject to

$$y_i - (\mathbf{w}^t\mathbf{x}_i + b) \le \varepsilon + \xi_{1i},$$
$$(\mathbf{w}^t\mathbf{x}_i + b) - y_i \le \varepsilon + \xi_{2i},$$
$$\xi_{1i}, \xi_{2i} \ge 0,\ i = 1,\dots,m$$

where $\boldsymbol{\xi}_1 = (\xi_{11},\dots, \xi_{1m})^t$ and $\boldsymbol{\xi}_2 = (\xi_{21},\dots, \xi_{2m})$ are vectors of slack variables. In a real-world situation, a function f that precisely approximates all pairs $(\mathbf{x}_i, y_i)$ may not exist, hence, similar to the soft margin classifier in subsection 2.2.1, we incorporated slack variables ξ_{1i}, ξ_{2i} for each pair $(\mathbf{x}_i, y_i)$ to capture their deviations out of ε-zone, as shown in Figure 2.6.

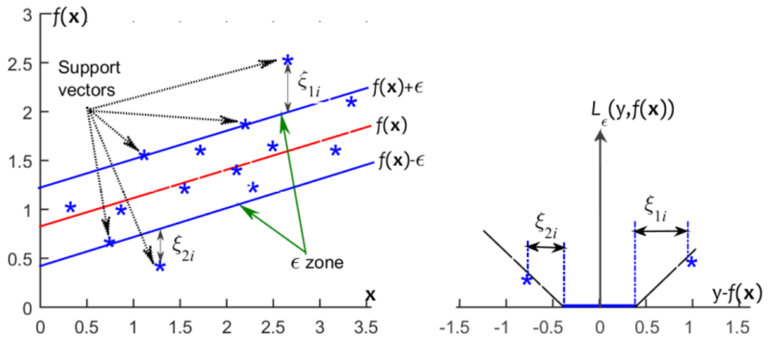


Figure 2.6 Geometrical interpretation of support vector regression with ε-insensitive loss

By considering the Lagrangian functions of the above QPP and then applying the Karush-Kuhn-Tucker (KKT) conditions leads to its following dual form

$$\min_{\boldsymbol{u}_1,\boldsymbol{u}_2\in R^m} \frac{1}{2}\sum_{i,j=1}^{m}(u_{1i}-\ u_{2i})\ \mathbf{x}_i^t\mathbf{x}_j(u_{1j}-u_{2j})\ +\ \varepsilon\sum_{i=1}^{m}(u_{1i}\ +u_{2i})-\sum_{i=1}^{m}(yi\ (u_{1i}-u_{2i}\ ) \tag{2.45}$$

$$\text{subject to}\sum_{i=1}^{m}(\,u_{1i}-u_{2i})\ =\ 0 \text{ and } 0\ \leq\ u_{1i},u_{2i}\ \leq\ c,\qquad i\ =1,\ldots,m$$

where $\mathbf{u}_1 = (u_{11},\ldots,u_{1m})^t$ and $\mathbf{u}_2 = (u_{21},\ldots,u_{2m})^t$ are vectors of Lagrangian multipliers.

As a result, $f\,(\mathbf{x})$ is stated using u_{1i} and u_{2i}.

$$f\ (\mathbf{x})\ =\sum_{i=1}^{m}(u_{1i}-u_{2i})\ \mathbf{x}_i^t\mathbf{x}+b \tag{2.46}$$

KKT conditions can be used to find out b.

The dual of SVR for nonlinear situations can be produced by applying the kernel method in (2.45).

$$\begin{aligned}\min_{\boldsymbol{u}_1,\boldsymbol{u}_2\in R^m} \frac{1}{2}&\sum_{i,j=1}^{m}(u_{1i}-\ u_{2i})\ k(\mathbf{x}_i^t,\mathbf{x}_j)(u_{1j}-u_{2j})\\ &+\ \varepsilon\sum_{i=1}^{m}(u_{1i}\ +u_{2i})-\sum_{i=1}^{m}(\,yi\ (u_{1i}-u_{2i}\ )\\ \text{subject to}&\sum_{i=1}^{m}(\,u_{1i}-u_{2i})\ =\ 0 \text{ and } 0\ \leq\ u_{1i},u_{2i}\ \leq\ c,\qquad i\ =1,\ldots,m\end{aligned} \tag{2.47}$$

where $k(\cdot,\cdot)$ is the kernel function of choice. For nonlinear SVR, the regression function $f(\mathbf{x})$ is (Cristianini & Shawe-Taylor, 2000; Vapnik, 2000)

$$f\ (\mathbf{x})\ =\sum_{i=1}^{m}(u_{1i}-u_{2i})\,k(\mathbf{x}_i,\mathbf{x})+b \tag{2.48}$$

2.3.2 Least Squares Support Vector Regression

In this part, we look at least squares SVR (LS-SVR), a simple reformulation of SVR (Suykens, 2000; Suykens et al., 2002b) for nonlinear cases. Instead of using an ε-insensitive loss, the LS-SVR uses a least square loss. Rather than solving a QPP as in SVR, LS-SVR needs to solve a system of linear equations deriving from KKT conditions. The equality restrictions are dealt with using LS-SVR, which is given by

$$\min_{(\boldsymbol{w},b,\boldsymbol{\xi})\in R^{n+1+m}} \frac{1}{2}\|\mathbf{w}\|^2 + c\sum_{i=1}^{m}\xi_i^2 \tag{2.49}$$

subject to $y_i = (\mathbf{w}^t\phi(\mathbf{x}_i) + b) + \xi_i, \quad i = 1,\ldots,m$

where $\phi(\cdot)$ denotes a mapping to a higher-dimensional feature space, c > 0 denotes a regularisation parameter, and $\boldsymbol{\xi} = (\xi_1,\ldots,\xi_m)^t$ denotes a vector of slack variables. After that, the Lagrangian with $\mathbf{u} = (u_1,\ldots,u_m)^t$ of (2.49) is built as

$$L(\mathbf{w}, b, \mathbf{u}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{c}{2}\sum_{i=1}^{m}\xi_i^2 + \sum_{i=1}^{m}u_i\,(y_i - (\mathbf{w}^t\phi(\mathbf{x}_i) + b) - \xi_i) \tag{2.50}$$

According to

$$\partial L/\partial\mathbf{w} = \mathbf{w} - \sum_{i=1}^{m}u_i\,\phi(\mathbf{x}_i) = 0, \partial L/\partial b = \sum_{i=1}^{m}u_i = 0, \partial L/\partial\xi_i = c\xi_i - u_i = 0, \partial L/\partial u_i$$
$$= y_i - \mathbf{w}^t\phi(\mathbf{x}_i) - b - \xi_i = 0,$$

after removing **w** and *b*, the KKT system is derived as follows:

$$\begin{bmatrix} 0 & \boldsymbol{e}^t \\ \boldsymbol{e} & I/C + K \end{bmatrix}\begin{bmatrix} b \\ \boldsymbol{u} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{y} \end{bmatrix}$$

where $\boldsymbol{y} = (y_1,\ldots y_m)^t$, **e** is a vector of ones of length *m*, *I* is an identity matrix of order *m* and *K* is a kernel matrix of order *m* whose (*i*, *j*)-*th* entry is $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. In this case, the nonlinear prediction function can be obtained to be

$$f(\boldsymbol{x}) = \sum_{i=1}^{m}u_i\,k(\boldsymbol{x}_i, \boldsymbol{x}) + b \tag{2.51}$$

Chapter 3

On Twin Bounded Support Vector Machine with Pinball Loss

3.1 Introduction

Since the support vector machine (SVM) has been proposed by Vapnik (Vapnik, 2000), it has emerged as a powerful machine learning tool for classification and regression owing to its theoretical analysis and excellent generalization performance in many fields of importance and applications such as face recognition, text categorization, bioinformatics (Guyon et al., 2002; Joachims,1998; Kim et al., 2010). The basic idea of SVM for classification is in finding an optimal decision boundary via maximizing the margin between the positive and negative classes of input pattern vectors (Vapnik, 2000). SVM is formulated as a convex quadratic programming problem (QPP) consisting of a regularization term and hinge loss based misclassification term (Vapnik, 2000).

Instead of constructing two parallel hyperplanes as in SVM, the concept of applying two non-parallel hyperplanes for determining the classifier was proposed firstly in (Mangasarian & Wild, 2006) leading to solving a pair of generalized eigenvalue problems. In (Jayadeva et. al., 2007), the authors proposed twin support vector machine (TWSVM) where two non-parallel hyperplanes were obtained by solving two smaller sized QPPs in comparison to SVM and as its enhanced version, which incorporates by the structural risk minimization concept, twin bounded SVM (TBSVM) was proposed in (Shao et al., 2011).

It is well-known that the decision boundary of SVM is determined by the misclassified input vectors known as support vectors (SVs) which leads to sparsity. As a consequence, SVs contaminated by feature noise will make the learned classifier corrupted and hence unreliable. This suggests that SVM is prone to noise and further unstable for re-sampling (Huang et al., 2014). In light of this, Huang et al. (2014) proposed pinball loss as a robust classification loss. As pinball is based on quantile distance, instead of maximizing the shortest distance between the classes, maximizing the quantile distance leads to novel Pin-SVM formulation having insensitivity to noise and stability of re-sampling (Huang et al., 2014). Also, pinball loss penalizes the patterns that are correctly classified and thereby brings them closer to the classifier resulting in low within-class scatter which is a preferred property of any reliable classifier. Though Pin-SVM possesses the above nice properties, it was noted that it lacks its sparseness. In order to attain it, ε-insensitive zone Pin-SVM is a new method that has been proposed also in

(Huang et al., 2014). As an optimal choice of the parameter ε, a modified $(\varepsilon_1, \varepsilon_2) -$ insensitive zone Pin-SVM is studied in (Rastogi et al., 2018). For the purpose of research an extension of pinball loss in place of hinge loss on twin parametric-margin support vector machine (TPMSVM) (Peng, 2011) for data classification, the interested reader is referred to (Xu et al., 2016). For the work on SVM with truncated pinball loss, can be found here (Yang & Dong, 2018).

In this chapter, inspired from the works of Pin-SVM and TBSVM, with the introduction of pinball loss for misclassification, we present a novel pinball loss based twin bounded support vector machine (Pin-TBSVM) having regularization term, scatter and misclassification loss term to enhance noise robustness. Pin-TBSVM is a non-parallel classifier where two kernel generated surfaces are constructed by solving quadratic programming problems (QPPs). Results of experiments on fourteen benchmark datasets reveal that the proposed method attains improved accuracy performance than the popular traditional methods considered for comparison clearly indicating the advantage of the proposed approach.

This chapter is organized as follows. Section 3.2 presents the proposed pinball TBSVM is formulated as two QPPs whose solutions are obtained by solving their duals. Numerical tests on (i) synthetic datasets with different types of noise/outliers; (ii) benchmark datasets with Gaussian noise, are performed in Section 3.3 while Section 3.4 concludes the paper.

3.2 Pinball Loss Twin Bounded Support Vector Machine (Pin-TBSVM)

With the objective of obtaining a robust classifier for noisy data, inspired by the research on Pin-SVM (Huang et al., 2014) and the problem formulation of TBSVM for hinge loss, we propose Pin-TBSVM as an extension of TBSVM for pinball loss in this section.

3.2.1 Linear Pin-TBSVM

The linear Pin-TBSVM seeks two non-parallel hyperplanes in R^n of the form

$$f_1(\boldsymbol{x}) = \boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0 \text{ and } f_2(\boldsymbol{x}) = \boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0$$

in which $\boldsymbol{w}_1, \boldsymbol{w}_2 \in R^n$ and $b_1, b_2 \in R$ are unknowns.

Following the problem formulation of TBSVM wherein replacing the hinge loss function by the pinball loss function, linear Pin-TBSVM is obtained leading to solving the following pair of optimization problems

$$\min_{\boldsymbol{w}_1, b_1} \ \frac{1}{2}c_3(\|\boldsymbol{w}_1\|^2 + b_1^2) + \frac{1}{2}\|A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1\|^2 + c_1 \sum_{j=m_1+1}^{m_1+m_2} L_{\tau_1}\big(\boldsymbol{x}_j, y_j, f_1(\boldsymbol{x}_j)\big)$$

and

$$\min_{\boldsymbol{w}_2,b_2} \ \frac{1}{2}c_4(\|\boldsymbol{w}_2\|^2+b_2^2)+\frac{1}{2}\|B\boldsymbol{w}_2+b_2\boldsymbol{e}_2\|^2+c_2\sum_{\mathrm{i=1}}^{m_1}L_{\tau_2}(\boldsymbol{x}_i,y_i,f_2(\boldsymbol{x}_i))$$

Or equivalently as QPPs

$$\begin{aligned}&\min_{\mathbf{w}_1,b_1,\boldsymbol{\xi}_2} \quad \frac{1}{2}c_3(||\mathbf{w}_1||^2+b_1^2)+\frac{1}{2}||A\mathbf{w}_1+b_1\mathbf{e}_1||^2+c_1\mathbf{e}_2^t\boldsymbol{\xi}_2\\ &\text{subject to} -(B\mathbf{w}_1+b_1\mathbf{e}_2)\geq \mathbf{e}_2-\boldsymbol{\xi}_2, \qquad -(B\mathbf{w}_1+b_1\mathbf{e}_2)\leq \mathbf{e}_2+\frac{1}{\tau_1}\boldsymbol{\xi}_2,\end{aligned} \tag{3.1a}$$

and

$$\begin{aligned}&\min_{\mathbf{w}_2,b_2,\boldsymbol{\xi}_1} \quad \frac{1}{2}c_4(||\mathbf{w}_2||^2+b_2^2)+\frac{1}{2}||B\mathbf{w}_2+b_2\mathbf{e}_2||^2+c_2\mathbf{e}_1^t\boldsymbol{\xi}_1\\ &\text{subject to } A\mathbf{w}_2+b_2\mathbf{e}_1\geq \mathbf{e}_1-\boldsymbol{\xi}_1, \qquad A\mathbf{w}_2+b_2\mathbf{e}_1\leq \mathbf{e}_1+\frac{1}{\tau_2}\boldsymbol{\xi}_1,\end{aligned} \tag{3.1b}$$

where for $k=1{,}2, c_k, c_{k+1}$ are regularization parameters, $\boldsymbol{\xi}_k \in R^{m_k}$ is a vector of slack variables, $\boldsymbol{e}_k \in R^{m_k}$ are the vector of ones and the parameter $0 \leq \tau_k \leq 1$.

One can notice that in the second term the intra-class compactness is minimized in 2-norm while, pinball is utilized for inter-class separation in the third term.

Remark 3.1 When $\tau_1=\tau_2=0$, the second constraints of (3.1) become $\boldsymbol{\xi}_k \geq \mathbf{0}$ and the problem reduces to TBSVM.

Remark 3.2 It is interesting to observe that when $\tau_1=\tau_2=1$, problem (3.1) becomes

$$\min_{\mathbf{w}_1,b_1\boldsymbol{\xi}_1,\boldsymbol{\xi}_2} \quad \frac{1}{2}c_3(||\mathbf{w}_1||^2+b_1^2)+\frac{1}{2}||\boldsymbol{\xi}_1||^2+c_1\|\boldsymbol{\xi}_2\|_1$$

$$\text{subject to } A\mathbf{w}_1+b_1\mathbf{e}_1=\boldsymbol{\xi}_1, (B\mathbf{w}_1+b_1\mathbf{e}_2)+\mathbf{e}_2=\boldsymbol{\xi}_2,$$

and

$$\min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}_1,\boldsymbol{\eta}_2} \quad \frac{1}{2}c_4(\|\boldsymbol{w}_2\|^2+b_2^2)+\frac{1}{2}\|\boldsymbol{\eta}_2\|^2+c_2\|\boldsymbol{\eta}_1\|_1$$

$$\text{subject to } B\boldsymbol{w}_2+b_2\boldsymbol{e}_2=\boldsymbol{\eta}_2, A\mathbf{w}_2+b_2\mathbf{e}_1-\mathbf{e}_1=\boldsymbol{\eta}_1$$

This is a special case of the general framework of non-parallel SVM formulation of (Mehrkanoon et al., 2014) using least-squares loss and L_1- norm loss and therefore it can be called as $LS-L_1$ loss non-parallel SVM.

By taking $\boldsymbol{z}_k=\begin{bmatrix}\boldsymbol{w}_k\\ b_k\end{bmatrix} \in R^{n+1}$ and the augmented matrices $G=[A \quad \boldsymbol{e}_1] \in R^{m_1\times(n+1)}$ and $H=[B \quad \boldsymbol{e}_2] \in R^{m_2\times(n+1)}$, problems (3.1a) and (3.1b) can be rewritten as

$$\begin{gathered}\min_{\mathbf{z}_1,\boldsymbol{\xi}_2} \quad \frac{1}{2}c_3||\mathbf{z}_1||^2 + \frac{1}{2}||G\mathbf{z}_1||^2 + c_1\mathbf{e}_2^t\boldsymbol{\xi}_2 \\ \text{subject to } H\boldsymbol{z}_1 \leq \boldsymbol{\xi}_2 - \boldsymbol{e}_2, \qquad -H\boldsymbol{z}_1 \leq \boldsymbol{e}_2 + \frac{1}{\tau_1}\boldsymbol{\xi}_2\end{gathered} \tag{3.2a}$$

and

$$\begin{gathered}\min_{\mathbf{z}_2,\boldsymbol{\xi}_1} \quad \frac{1}{2}c_4||\mathbf{z}_2||^2 + \frac{1}{2}||H\mathbf{z}_2||^2 + c_2\mathbf{e}_1^t\boldsymbol{\xi}_1 \\ \text{subject to} - G\mathbf{z}_2 \leq \boldsymbol{\xi}_1 - \boldsymbol{e}_1,\ \mathrm{G}\mathbf{z_2} \leq \mathbf{e}_1 + \frac{1}{\tau_2}\boldsymbol{\xi}_1\end{gathered} \tag{3.2b}$$

The primal problems (3.2a) and (3.2b) will be solved by constructing their duals. Consider (3.2a), with the introduction of Lagrange multipliers $\boldsymbol{u}_2, \boldsymbol{v}_2$ in R^{m_2}, the Lagrangian function corresponding to (3.2a) becomes

$$\begin{gathered}L_1(\boldsymbol{z}_1, \boldsymbol{\xi}_2, \boldsymbol{u}_2, \boldsymbol{v}_2) = \frac{1}{2}c_3\boldsymbol{z}_1^t\boldsymbol{z}_1 + \frac{1}{2}\boldsymbol{z}_1^tG^tG\boldsymbol{z}_1 + \boldsymbol{u}_2^t(H\boldsymbol{z}_1 - \boldsymbol{\xi}_2 + \boldsymbol{e}_2) \\ +\boldsymbol{v}_2^t(-H\boldsymbol{z}_1 - \boldsymbol{e}_2 - \frac{1}{\tau_1}\boldsymbol{\xi}_2)\end{gathered} \tag{3.3}$$

Using the Karush-Kuhn-Tucker (KKT) necessary optimality conditions

$\partial L_1/\partial \boldsymbol{z}_1 = \mathbf{0} \Rightarrow (c_3I + G^tG)\boldsymbol{z}_1 = -H^t(\boldsymbol{u}_2 - \boldsymbol{v}_2),\ \partial L_1/\partial \boldsymbol{\xi}_2 = \mathbf{0} \Rightarrow \boldsymbol{u}_2 + \frac{\boldsymbol{v}_2}{\tau_1} = c_1\boldsymbol{e}_2$

and eliminating the primary variables $\boldsymbol{z}_1, \boldsymbol{\xi}_2$ in (3.3), the dual of (3.2a) becomes

$$\begin{gathered}\min_{\boldsymbol{u}_2,\boldsymbol{v}_2 \in R^{m_2}} \frac{1}{2}(\boldsymbol{u}_2 - \boldsymbol{v}_2)^tH(c_3I + G^tG)^{-1}H^t(\boldsymbol{u}_2 - \boldsymbol{v}_2) - \boldsymbol{e}_2^t(\boldsymbol{u}_2 - \boldsymbol{v}_2) \\ \text{subject to } \boldsymbol{u}_2 + \boldsymbol{v}_2/\tau_1 = c_1\boldsymbol{e}_2 \text{ and } \boldsymbol{u}_2 \geq \mathbf{0}, \boldsymbol{v}_2 \geq \mathbf{0}\end{gathered} \tag{3.4}$$

Since $\boldsymbol{u}_2 + \boldsymbol{v}_2/\tau_1 = c_1\boldsymbol{e}_2$, we have $\boldsymbol{u}_2 = c_1\boldsymbol{e}_2 - \boldsymbol{v}_2/\tau_1 \geq \mathbf{0}$ and $\boldsymbol{u}_2 - \boldsymbol{v}_2 = c_1\boldsymbol{e}_2 - (1 + 1/\tau_1)\boldsymbol{v}_2$.

Putting this results in (3.4) and further taking $Q_1 = H(c_3I + G^tG)^{-1}H^t$, problem (3.4) becomes

$$\begin{gathered}\min_{\boldsymbol{v}_2 \in R^{m_2}} \frac{1}{2}(1 + 1/\tau_1)\boldsymbol{v}_2^tQ_1\boldsymbol{v}_2 - \boldsymbol{e}_2^t(2c_1Q_1 - I)\boldsymbol{v}_2 \\ \text{subject to } \mathbf{0} \leq \boldsymbol{v}_2 \leq c_1\tau_1\boldsymbol{e}_2.\end{gathered} \tag{3.5a}$$

Similarly, by introducing Lagrange multipliers $\boldsymbol{u}_1, \boldsymbol{v}_1$ in R^{m_1}, the dual of (3.2b) becomes

$$\begin{gathered}\min_{\boldsymbol{v}_1 \in R^{m_1}} \frac{1}{2}(1 + 1/\tau_2)\boldsymbol{v}_1^tQ_2\boldsymbol{v}_1 - \boldsymbol{e}_1^t(2c_2Q_2 - I)\boldsymbol{v}1 \\ \text{subject to } \mathbf{0} \leq \boldsymbol{v}_1 \leq c_2\tau_2\boldsymbol{e}_1,\end{gathered} \tag{3.5b}$$

satisfying the necessary optimality conditions $(c_4 I + H^t H)\boldsymbol{z}_2 = G^t(\boldsymbol{u}_1 - \boldsymbol{v}_1)$ and $\boldsymbol{u}_1 - \boldsymbol{v}_1 = c_2\boldsymbol{e}_1 - (1 + 1/\tau_2)\boldsymbol{v}_1$ where $Q_2 = G(c_4 I + H^t H)^{-1} G^t$.

Remark 3.3. In the derivation of our formulation (3.5), it is assumed that $(\tau_1, \tau_2) \neq (0,0)$. Using the solutions of (3.5a) and (3.5b), one can obtain

$$\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} = (c_3 I + G^t G)^{-1} H^t ((1 + 1/\tau_1)\boldsymbol{v}_2 - c_1 \boldsymbol{e}_2),$$
$$\begin{bmatrix} \boldsymbol{w}_2 \\ b_2 \end{bmatrix} = (c_4 I + H^t H)^{-1} G^t (c_2 \boldsymbol{e}_1 - (1 + 1/\tau_2)\boldsymbol{v}_1) \tag{3.6}$$

and accordingly, the linear Pin-TBSVM's decision function is provided by

$$f(\boldsymbol{x}) = sign\left(\frac{\boldsymbol{w}_1^t \boldsymbol{x} + b_1}{||\boldsymbol{w}_1||} + \frac{\boldsymbol{w}_2^t \boldsymbol{x} + b_2}{||\boldsymbol{w}_2||}\right).$$

3.2.2 Non-linear Pin-TBSVM

The Pin-TBSVM for the linear case can be extended to non-linear problem by seeking two kernel generated surfaces of the form $f_1(\boldsymbol{x}) = K(\boldsymbol{x}^{\boldsymbol{t}}, C^t)\boldsymbol{w}_1 + b_1 = 0 \; and \; f_2\,(\boldsymbol{x}) = K(\boldsymbol{x}^{\boldsymbol{t}}, C^t)\boldsymbol{w}_2 + b_2 = 0$. They are obtained by solving the pair of QPPs

$$\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}_2} \quad \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + \frac{1}{2} ||K(A, C^t)\boldsymbol{w}_1 + b_1 \boldsymbol{e}_1||^2 + c_1 \boldsymbol{e}_2^t \boldsymbol{\xi}_2$$

$$\text{subject to} - (K(B, C^t)\boldsymbol{w}_1 + b_1 \boldsymbol{e}_2) \geq \boldsymbol{e}_2 - \boldsymbol{\xi}_2,$$

$$-(K(B, C^t)\boldsymbol{w}_1 + b_1 \boldsymbol{e}_2) \leq \boldsymbol{e}_2 + \frac{1}{\tau_1}\boldsymbol{\xi}_2,$$

and

$$\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\xi}_1} \quad \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + \frac{1}{2} ||K(B, C^t)\boldsymbol{w}_2 + b_2 \boldsymbol{e}_2||^2 + c_2 \boldsymbol{e}_1^t \boldsymbol{\xi}_1$$

$$\text{subject to } K(A, C^t)\boldsymbol{w}_2 + b_2 \boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\xi}_1,$$

$$K(A, C^t)\boldsymbol{w}_2 + b_2 \boldsymbol{e}_1 \leq \boldsymbol{e}_1 + \frac{1}{\tau_2}\boldsymbol{\xi}_1.$$

By defining $G = [K(A, C^t) \quad \boldsymbol{e}_1] \in R^{m_1 \times (m+1)}$ and $H = [K(B, C^t) \quad \boldsymbol{e}_2] \in R^{m_2 \times (m+1)}$and proceeding as in the linear problem, a pair of dual problems of the form (3.5a), (3.5b) is solved. Finally, we obtain the decision function

$$f(\boldsymbol{x}) = sign\left(\frac{K(\boldsymbol{x}^t, C^t)\boldsymbol{w}_1 + b_1}{||\boldsymbol{w}_1||} + \frac{K(\boldsymbol{x}^t, C^t)\boldsymbol{w}_2 + b_2}{||\boldsymbol{w}_2||}\right)$$

where the unknowns $\boldsymbol{w}_k$ and b_k are given by (3.6).

Notice that Pin-TBSVM requires the computation of the inverse of the matrices $(c_3 I + G^t G)$ and $(c_4 I + H^t H)$. However, using Sherman-Morrison-Woodbury (SMW) identity, we get $(c_3 I + G^t G)^{-1} = (I - G^t (c_3 I + G G^t)^{-1} G)/c_3$ and similarly $(c_4 I + H^t H)^{-1} = (I - H^t (c_4 I + H H^t)^{-1} H)/c_4$. This requires the inverse of matrices $(c_3 I + G G^t)$ and $(c_4 I + H H^t)$ of lower orders m_1 and m_2 than the original matrices of larger order (m+1).

3.3 Experimental Results

In this section, we compare the performance of our proposed Pin-TBSVM with SVM, TBSVM and Pin-SVM on 14 popular benchmark datasets from the UCI machine learning repository available at http://www.ics.uci.edu/~mlearn. All computations were performed using MATLAB R2015a on a PC running on Windows 10 OS with 64 bit, 3.40 GHz Intel®core™ i7 processor with 8 GB of RAM. We solved QPPs using the MOSEK optimization toolbox for MATLAB available at http://www.mosek.com. The popular Gaussian kernel $k(\boldsymbol{x}, \boldsymbol{z}) = exp(-||\boldsymbol{x} - \boldsymbol{z}||^2 / 2\sigma^2)$ is considered where $\sigma > 0$ is a parameter. The optimal parameter values are determined by a tenfold cross-validation methodology where the penalty parameters c_1, c_2, c_3, c_4 and the kernel parameter σ assume values from the set {2^i| i = -9, -8, -7,…,9} whereas τ_1, τ_2 vary from {0.1, 0.2, 0.5, 1}. In fact, by randomly splitting the dataset into ten equal parts or folds and taking one of the folds as the validation set and the rest for training, the parameters showing the highest averaged accuracy are selected and they are used to determine the classifier using the full training set. To ease the computational cost, we assumed $c_1 = c_3, c_2 = c_4$, $\nu_1 = \nu_2$ and $\tau = \tau_1 = \tau_2$.

We test the effectiveness of Pin-TBSVM in terms of classification accuracy and training time by performing experiments on the datasets under noiseless and noisy settings. For all the datasets considered, their feature values are normalized to lie in [0, 1]. As noise setting, we let each feature value be corrupted by Gaussian noise with zero-mean in which noise variance to feature variance ratio (r) is defined to be r=0 (i.e., noise-free), r=0.05 and r=0.1 (Huang et al., 2014). The same noise contaminated both training and test vectors.

The experimental results showing the accuracy of Gaussian kernel are presented in Table 3.1. In most of the cases, in terms of classification accuracy performance of Pin-TBSVM is better than SVM, TBSVM and Pin-SVM. In fact, the highest accuracy was yielded by the proposed method in 33 times out of 42 cases (14 datasets x 3 noise levels). This indicates the superiority of Pin-TBSVM. In general, the precision of a system reduces as the degree of noise rises. In addition, algorithms were ranked for each dataset so that rank 1 is assigned for the

Table 3.1 Accuracy comparison of Pin-TBSVM with SVM, TBSVM and Pin-SVM on real-world benchmark datasets with noise level r. Gaussian kernel was employed.

Dataset (Total size)	Level of noise	SVM Time (Rank)	TBSVM Time (Rank)	Pin-SVM Time (Rank)	Pin-TBSVM Time (Rank)
Cleveland (297×13)	*r*=0	84.50±8.07 0.1334 (3)	**85.51±6.54** 0.0742 (1)	84.25±8.55 0.1428 (4)	85.16±7.30 0.1997 (2)
	r=0.05	84.30±6.41 0.1321 (3)	84.20±6.37 0.0758 (4)	84.34±6.57 0.1444 (2)	**84.51±7.04** 0.1430 (1)
	r=0.1	83.86±7.94 0.1343 (3)	83.80±7.87 0.0739 (4)	84.88±5.97 0.1439 (2)	**85.52±5.91** 0.1430 (1)
Breast-cancer (683×9)	*r*=0	97.26±1.16 0.8155 (4)	97.51±1.22 0.6568 (3)	97.95±1.57 1.1370 (2)	**98.24±1.15** 0.7259 (1)
	r=0.05	**97.51±1.95** 0.8110 (1.5)	**97.51±1.32** 0.6366 (1.5)	97.06±1.95 1.1362 (4)	97.37±2.14 0.6679 (3)
	r=0.1	96.92±1.28 0.8092 (3.5)	96.92±1.12 0.6577 (3.5)	97.36±1.93 1.1374 (2)	**97.95 ±1.95** 0.6256 (1)
WPBC (194×33)	*r*=0	**82.47±9.02** 0.0708 (1)	81.80±9.99 0.0500 (2)	81.75±7.88 0.0767 (3)	78.02±13.68 0.0921 (4)
	r=0.05	80.89±9.56 0.0699 (4)	80.94±9.89 0.0499 (3)	81.80±9.81 0.0747 (2)	**82.94 ±5.63** 0.0937 (1)

	r=0.1	79.44±6.72 0.0678 (4)	80.36±7.65 0.0497 (3)	80.44±5.81 0.0704 (2)	**81.5±8.93** 0.0872 (1)
Sonar (208×60)	*r*=0	90.45±5.22 0.0717 (4)	90.87±4.87 0.0593 (2)	90.80±5.30 0.0740 (3)	**91.83±5.52** 0.1073 (1)
	r=0.05	89.44±6.63 0.0713 (4)	89.50±5.31 0.0570 (3)	89.62±6.30 0.0749 (2)	**91.33±7.72** 0.0849 (1)
	r=0.1	88.47±7.62 0.0718 (3.5)	88.47±7.01 0.0597 (3.5)	88.58±6.70 0.0741 (2)	**89.5±6.63** 0.0925 (1)
Heart statlog (270×13)	*r*=0	84.41±4.43 0.1153 (4)	84.44±5.00 0.0867 (3)	**84.81±4.93** 0.1168 (1.5)	**84.81±3.24** 0.1418 (1.5)
	r=0.05	84.07±5.53 0.1149 (4)	84.20±6.22 0.0870 (3)	84.55±10.39 0.1180 (2)	**85.55±4.43** 0.1144 (1)
	r=0.1	83.44±7.36 0.1142 (3.5)	83.44±7.48 0.0866 (3.5)	83.54±5.46 0.1165 (2)	**83.70±3.98** 0.1207 (1)
Australian Credit (690×14)	*r*=0	86.52±3.05 0.8146 (4)	87.82±2.96 0.4741 (2)	87.10±2.98 0.9605 (3)	**88.69±3.97** 0.7576 (1)
	r=0.05	86.08±2.99 0.8144 (4)	86.25±2.98 0.4493 (3)	86.45±3.43 0.9901 (2)	**86.95±4.42** 0.6373 (1)
	r=0.1	86.95±3.97 0.8076	86.85±3.64 0.4868	87.00±2.97 0.9695	**87.97±3.13** 0.6502

		(3)	(4)	(2)	(1)
Votes (435×16)	*r*=0	96.79±3.59 0.2907 (2)	96.20±2.48 0.1351 (4)	96.32±2.98 0.2915 (3)	**97.47±1.98** 0.3411 (1)
	r=0.05	96.52±2.46 0.2847 (2)	95.51±2.43 0.1386 (3)	96.50±3.07 0.2900 (4)	**97.70±1.53** 0.2959 (1)
	r=0.1	96.09±2.50 0.2834 (3)	95.08±1.88 0.1382 (4)	96.18±1.55 0.2869 (2)	**96.98±2.90** 0.2737 (1)
Haberman (306×3)	*r*=0	75.09±9.99 0.1591 (4)	75.78±7.62 0.1190 (3)	75.81±7.54 0.2552 (2)	**76.75±9.15** 0.1740 (1)
	r=0.05	74.79±8.20 0.1574 (4)	74.99±6.43 0.1079 (3)	75.17±7.41 0.2544 (2)	**76.75±10.28** 0.1519 (1)
	r=0.1	73.88±7.36 0.1568 (3)	73.92±5.67 0.1099 (2)	**74.20±6.54** 0.2513 (1)	73.81±9.91 0.1544 (4)
Ionosphere (351×33)	*r*=0	93.30±2.70 0.2110 (3)	92.87±2.03 0.1108 (4)	**94.59±1.33** 0.2265 (1)	94.03±5.59 0.2223 (2)
	r=0.05	91.75±5.74 0.2101 (4)	92.17±2.10 0.1118 (3)	92.87±2.03 0.2206 (2)	**93.75±5.62** 0.2134 (1)
	r=0.1	90.15±4.71 0.2010 (4)	90.92±3.46 0.1142 (3)	91.46±5.85 0.2148 (2)	**91.75±7.98** 0.2098 (1)
WDBC	*r*=0	98.06±2.10 0.5907	98.08±2.34 0.5252	97.98±3.78 1.0744	**99.47±0.84** 0.5579

(569×30)		(3)	(2)	(4)	(1)
	r=0.05	97.64±1.43 0.5871 (3)	97.71±1.69 0.4484 (2)	**97.78±1.69** 1.1068 (1)	96.48 ±1.65 0.5087 (4)
	r=0.1	97.07±1.93 0.5850 (3)	96.62±1.98 0.4415 (4)	97.20±1.86 1.0717 (2)	**98.06±1.53** 0.4722 (1)
Transfusion (748×4)	*r*=0	79.81±2.64 1.1665 (4)	80.05±2.70 0.7954 (3)	80.10±2.80 2.4669 (2)	**80.11±15.32** 1.0554 (1)
	r=0.05	78.63±3.78 1.1778 (3)	**78.75±3.26** 0.7914 (1)	78.65±5.26 2.4828 (2)	76.23±15.29 0.7650 (4)
	r=0.1	77.20±5.13 1.1623 (3)	**77.40±4.01** 0.7924 (1)	77.25±3.96 2.4632 (2)	76.23±15.53 0.7151 (4)
Pima Indians (768×8)	*r*=0	78.13±5.98 1.1616 (4)	78.30±4.63 0.8044 (3)	78.50±3.56 2.8370 (2)	**79.55±4.41** 0.8842 (1)
	r=0.05	77.22±4.46 1.1598 (4)	77.25±5.16 0.8055 (3)	77.60±6.06 2.4607 (2)	**78.25±5.66** 1.0125 (1)
	r=0.1	76.94±6.30 1.1655 (4)	76.97±5.84 0.8082 (3)	77.12±4.34 2.3969 (2)	**77.99±3.75** 0.8538 (1)
German (1000×24)	*r*=0	76.40±6.37 1.9444 (4)	76.91±3.92 1.7613 (3)	77.10±1.86 4.1229 (2)	77.30±3.56 1.6584 (1)
	r=0.05	75.90±4.12 1.9611	76.00±4.63 1.7189	76.30±5.59 4.2436	**77.9±3.10** 1.4645

		(4)	(3)	(2)	(1)
	r=0.1	75.30±5.85 1.9843 (3.5)	75.30±5.85 1.7788 (3.5)	76.00±5.86 4.4651 (2)	**76.60±3.16** 1.4147 (1)
CMC (1473 X 9)	*r*=0	75.15±2.74 5.8026 (3.5)	75.15±4.91 3.3092 (3.5)	75.20±5.23 11.7514 (2)	**75.70±4.52** 3.4301 (1)
	r=0.05	74.98±4.68 5.6304 (4)	75.01±5.47 3.3270 (3)	75.20±3.37 9.9968 (2)	**77.53±2.37** 3.1488 (1)
	r=0.1	75.00±3.65 5.8822 (4)	75.04±4.26 3.3445 (3)	75.08±4.16 10.3863 (2)	**75.36±4.68** 3.2903 (1)

best accuracy performer. The results are given in Table 3.1. In Table 3.2, we have reported their average ranks. Note that a smaller average rank means better prediction performance. At all levels of noise, the best prediction performance is achieved by Pin-TBSVM and is followed by Pin-SVM, TBSVM and lastly by SVM. This clearly shows the superiority of Pin-TBSVM. Enhanced performance by Pin-TBSVM and Pin-SVM in comparison to hinge loss SVMs on datasets corrupted by noise illustrates the robustness of pinball loss based SVMs. On learning time, though TBSVM shows the best result, our proposed method is comparable with TBSVM. As expected, Pin-SVM takes more time than the rest for training.

Table 3.2 Average ranks on the accuracy for Gaussian kernel on real-world datasets with noise level r.

Level of noise	SVM	TBSVM	Pin-SVM	Pin-TBSVM
r=0	3.39	2.75	2.46	1.39
r=0.05	3.46	2.75	2.21	1.57
r=0.1	3.42	3.21	1.92	1.42

In continuation of the above analysis, we verify the comparative performance of the algorithms statistically by applying the non-parametric Friedman test followed by Nemenyi post-hoc test reported in (Demsar, 2006). Under the assumption of the null hypothesis that all the algorithms are equivalent, the Friedman statistic distributed according to $\chi_F^2 = \frac{12N}{k(k+1)}[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}]$ distribution with $(k-1)$ degrees of freedom and a better statistic $F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$ distributed according to F −distribution with $((k-1),(k-1)(N-1))$ degrees of freedom are computed where $k =$ the number of algorithms, $N =$ the number of datasets and $R_j =$ the average rank of the jth algorithm. When the null hypothesis is rejected, a pair-wise comparison of the algorithms is performed with the Nemenyi post-hoc test by computing the critical difference (CD) (Demsar, 2006).

From Table 3.2, using the average ranks for $r = 0$ with $k = 4$ and $N = 14$:

$$\chi_F^2 = \frac{12 \times 14}{4 \times 5}(3.39^2 + 2.75^2 + 2.46^2 + 1.39^2 - \frac{4 \times 5^2}{4}) \approx 17.1217,$$

$F_F = \frac{13 \times 17.1217}{14 \times 3 - 17.1217} \approx 8.9468$ with $(3,3 \times 13) = (3,39)$ degrees of freedom. The critical value of $F(3,39)$ at the level of significance $\alpha = 0.05$ is 2.8451. However, $F(3,39) = 2.8451$, we reject the null hypothesis. So, we proceed for pair-wise comparison using Nemenyi post-hoc test. From (Demsar, 2006), the value of CD at $p = 0.10$ is 1.1179. If the average ranks of two algorithms differ at least by CD then their performance differs significantly (Demsar, 2006). Thus, from Table 3.2 and when $r = 0$, the difference between the average ranks of: (i). Pin-TBSVM and Pin-SVM is: 2.46-1.39=1.07 < 1.1179 (CD), the post-hoc test is not powerful enough to detect any significant difference between the algorithms; (ii). Pin-TBSVM and the best of SVM and TB-SVM is: 2.75-1.39=1.36 > 1.1179, we conclude that the performance of Pin-TBSVM is better than SVM and TBSVM.

Similarly, with $r = 0.05$, we calculate $\chi_F^2 = 15.8180$ and $F_F = 7.8541$. Here, we see that $F_F > F(3,39) = 2.8451$ and hence we reject the null hypothesis. For pair-wise comparison using Nemenyi post-hoc test, as the difference between the average ranks of : (i). Pin-TBSVM and Pin-SVM is: 2.21- 1.57 = 0.64 < 1.1179, there is no significant difference between the algorithms according to the post-hoc test; (ii). Pin-TBSVM and the best of SVM and TB-SVM is: 2.75 - 1.57 = 1.18 > 1.1179, we say that the Pin-TBSVM outperforms SVM and TBSVM in terms of performance.

As our final algorithm statistical comparison, for $r = 0.1$, we compute $\chi_F^2 = 22.7077$ and $F_F = 15.3015$. Since $F_F > F(3,39)$, we reject the null hypothesis and so perform the post-hoc test. Again from Table 3.2, the difference between the average ranks of : (i). Pin-TBSVM and Pin-SVM are 1.92 – 1.42 = 0.5 < 1.1179, we say that the test is not powerful enough to determine any significant difference between the algorithms; (ii). Pin-TBSVM and the best of

SVM and TB-SVM is: 3.21 - 1.42 = 1.79 > 1.1179, we conclude that the performance of Pin-TBSVM is superior to SVM and TBSVM.

From the statistical tests, we say that Pin-TBSVM is more efficient than SVM and TBSVM and at the same time, it compares favourably with Pin-SVM.

In summary, we conclude that Pin-TBSVM is an efficient classification method for datasets corrupted by feature noise.

3.4 Conclusions

In this chapter, we presented a robust twin bounded support vector machine with pinball loss (Pin-TBSVM) for feature noise affected datasets. Pin-TBSVM is a non-parallel classifier where kernel generated surfaces were determined as the solutions of quadratic programming problems. Experimental results on several benchmark datasets show that the proposed method achieves improved accuracy performance than the popular traditional methods. Though Pin-TBSVM is a simple and efficient learning method, it loses sparsity. An increase in the number of parameters is a concern and the selection of their optimal values is a practical problem that needs attention. The proposed method's application to multi-category classification problems is interesting and worth investigating.

Chapter 4

Robust Twin Bounded Support Vector Machine with Pinball Loss for Data Classification

4.1 Introduction

In this chapter, a novel L_1 −norm based twin bounded support vector machine with pinball loss having regularization term, scatter loss and misclassification loss is proposed to enhance robustness in the presence of feature noise and outliers. Unlike in twin bounded support vector machine (TBSVM), pinball is used as the misclassification loss in place of hinge loss to reduce noise sensitivity. To further boost robustness, the scatter loss of the class of vectors is minimized using L_1 −norm. As an equivalent problem in simple form, a pair of quadratic programming problems (QPPs) is constructed (L1-Pin-TBSVM) with m variables where m is the number of training vectors. Unlike TBSVM, the proposed L1-Pin-TBSVM is free from inverse kernel matrix and the non-linear problem can be obtained directly from its linear formulation by applying the kernel trick. The efficacy and robustness of L1-Pin-TBSVM have been demonstrated by experiments performed on synthetic and UCI datasets in the presence of noise.

Over the past decades, the support vector machine (SVM) introduced by Vapnik (Cortes, 1995; Vapnik, 2000) has become a popular machine learning tool for classification problems because of its solid mathematical foundation and excellent performance on many real-world problems of applications such as face recognition, text categorization, bioinformatics (Guyon et al., 2002; Joachims, 1998; Kim et al., 2010). The classical SVM finds an optimal hyperplane separating between the positive and negative class of input vectors with maximum margin (Cortes & Vapnik 1995; Vapnik, 2000). This leads to solving a convex quadratic programming problem (QPP) (Vapnik, 2000; Cristianini & Shawe-Taylor, 2000) consists of the regularization and misclassification error terms where the error is computed using hinge loss.

Though SVM provides a sparse and global solution, its main difficulty is the high computational cost in solving the QPP on large problems, i.e., its learning time complexity is $O(m^3)$ where m represents the number of training vectors. To get around this difficulty, fast SVM learning algorithms that are based on methods of optimization such as decomposition method (Osuna, 1997), sequential minimal optimization (SMO) (Platt, 1999), dual coordinate descent (DCD) (Hsieh et al., 2008) and successive over-relaxation (SOR) (Mangasarian & Musicant, 1999) have been reported in the literature. Inspired from the work of generalized eigenvalue proximal support vector machine (GEPSVM) of (Mangasarian & wild, 2006), Jayadeva et al. (2007) proposed a twin support vector machine (TWSVM), a variant of SVM as an alternative approach, where a pair of non-parallel hyperplanes were constructed by solving two QPPs of smaller size. This strategy makes TWSVM four times faster than SVM (Jayadeva et al., 2007). Because of its low computational cost and better generalization performance than SVM, different forms of TWSVM and its extensions have been proposed in the literature (Balasundaram et al., 2017; Kumar & Gopal 2009; Peng, 2010; Peng, 2011; Peng et al., 2016; Shao et al., 2011).

The other issue with SVM is that since the margin is measured by the extreme values, once they are contaminated by noise, the resulting classifier will be susceptible to feature noise around the classification boundary and further unstable for re-sampling (Haung et al., 2014).To address this issue, Huang et al. (2014) introduced pinball loss in the SVM classification problem where the idea of maximizing the shortest distance between the two classes is changed into maximizing the quantile distance, resulting in the novel formulation of pinball loss SVM (Pin-SVM). In contrast to SVM, Pin-SVM penalizes the correctly classified training vectors which makes the model in minimizing the scatters around the parallel planes. Though Pin-SVM possesses a small within-class scatter property, it was noticed that it loses its sparsity. In order to attain that, $\varepsilon -$insensitive zone Pin-SVM is proposed in (Huang et al., 2014). As an optimal insensitive zone SVM with pinball loss, a modified $(\varepsilon_1, \varepsilon_2) -$ insensitive zone Pin-SVM is studied in (Rastogi et al., 2018). As an extension on the study of pinball loss to TWSVM where pinball loss is employed to twin parametric-margin support vector machine (TPMSVM) (Peng et al., 2011) for data classification, Xu et al. (2016) proposed pinball loss twin support vector machine (Pin-TSVM). By measuring the margin between the two classes with expectile value and using its relation to asymmetric squared pinball loss, asymmetric least squares support vector machine (aLS-SVM) is proposed in (Haung et al., 2014). In a recent work on L2-norm pinball based twin bounded support vector machine, the reader is referred to (Prasad & Balasundaram, 2021).

In real-world applications, data samples are often contaminated by different types of noise. While small perturbations resulted usually due to erroneous measurements are referred as feature noise, the samples which are significantly different from majority of the others such as wrongly labelled samples are defined as outliers (Frenay & Verleysen, 2014). Robustness to outliers is an important problem of study in pattern classification. It is well-known that the presence of outliers can ruin the resulting classifier and hence the learned classifier can become unreliable. A popular approach for constructing robust learning models is in developing algorithms based on robust optimization such as the application of a re-weighted strategy where different weights are assigned to the training samples according to their residuals (Suykens et al., 2002). Samples with large residuals may be assumed as outliers and consequently small weights are assigned to them so that their effect can be reduced while learning. The main difficulty in this approach is usually it requires additional computational costs for training than SVM. The other approach is in designing robust loss functions so that they resist the effect of outliers. Though L_2 −norm distance is insensitive to feature noise, the squared operation can significantly influence large outliers resulting poor classification performance by the learning model (Mehrkanoon et al., 2014). The popular TWSVM (Jayadeva et al., 2007) and its variant TBSVM (Shao et al., 2011) are sensitive to outliers because they implement L_2 −norm distance. It is argued that both the hinge loss and pinball loss are unbounded and therefore they are not enough robust to outliers (Yang & Dong, 2018). In keeping view of this, truncated non-convex loss functions are proposed and robust models are constructed in (Yang & Dong, 2018; Haung et al., 2014, Shen et al., 2017). However, non-convexity introduces difficulty in optimization. As a promising alternative way to address outliers, many researchers in the literature of machine learning have often preferred L_1 −norm measure (Gao et al., 2011; Yan et al., 2018; Yan et al., 2019). Finally, on the recent work on least-squares TBSVM based on L_1 −norm (L1-LSTBSVM) where inequality constraints are replaced by equalities, we refer to (Yan et al., 2018).

With the aim of obtaining a learning model robust to both feature noise and outliers, a non-parallel support vector machine under the framework involving a regularization term, a scatter loss and misclassification loss using robust loss functions is proposed in this work. Like TBSVM, the presence of the regularization term in the objective function has the advantage that the structural risk is minimized. Since L_1 −norm distance makes the learning machines robust to outliers and pinball has the advantage of enhanced robustness to feature noise, with inspiration from the works of (Shao et al., 2011; Haung et al., 2014, Xu et al., 2016), a new twin bounded support vector machine classifier with pinball loss is developed as a non-parallel SVM where the within-class scatter is minimized in L_1 −norm and pinball is used for misclassification error. Our problem is further simplified into an equivalent problem of solving a pair of SVM-type constrained QPPs in the dual space (L1-Pin-TBSVM) in *m* variables where *m* represents the number of training vectors. With the dual solution, the end classifier is obtained. Experiments were performed on *Crossplanes* dataset having two or four outliers, two-moon dataset and eleven benchmark datasets with different levels of noise. Their results in comparison with the state of the art variants of SVM show the robustness, effectiveness and suitability of L1-Pin-TBSVM.

The following are the primary contributions of this work:

- With the aim of obtaining enhanced generalization performance for feature noise and outliers, a novel twin bounded support vector machine with pinball loss for data classification is proposed.

- As an efficient way to reduce the influence of outliers, the robust L_1 −norm is used for minimizing the within-class scatter and to bring noise insensitivity pinball is introduced into the classification.
- Our formulation leads to a pair of QPPs with box constraints (L1-Pin-TBSVM) having *m* number of variables in the dual space and free of matrix inversion terms where *m* represents the number of training vectors.
- As in the classical SVM, the duals for non-linear kernel can be derived directly from their linear formulations by applying kernel trick.
- the number of variables in Pin-TSVM and L1-Pin-TBSVM is equal to the number of training vectors if the training set contains the same number of data points from both classes.
- Empirical findings based on synthetic datasets, with outliers and feature noise, and in addition on sixteen benchmark datasets with three distinct amounts of noise in the datasets, confirm the efficiency and superiority.

In this work, all vectors are taken as column vectors. For any vector $\boldsymbol{x} = (x_1, \ldots, x_n)^t \in R^n$, let $\boldsymbol{x}^t$ be its transpose. Its L_1 −norm and L_2 −norm is denoted by $||\boldsymbol{x}||_1$ and $||\mathbf{x}||$ respectively. $\mathbf{0}$ and **e** correspondingly denote column vectors of zeros and ones of appropriate dimensions and *I* stands for an identity matrix of appropriate size.

The rest of the chapter is organized as follows. In Section 4.2, TBSVM in L_1 −norm with pinball loss is proposed whose properties of noise insensitivity and scatter minimization are verified and further as an equivalent problem in simple form, a pair of QPPs in the dual space is derived. In Section 4.3, a comparative study of the proposed method with TBSVM, Pin-SVM and Pin-TSVM is presented. Numerical experiments are detailed and their results are reported in Section 4.4 while Section 4.5 concludes the chapter.

4.2 Robust Twin Bounded Support Vector Machine with Pinball Loss

Motivated by the works of TBSVM and Pin-TSVM, we develop a novel L_1 −norm based twin bounded SVM with pinball loss (Pin-TBSVM) to enhance the robustness in this section. In addition to noise insensitivity, we will verify that Pin-TBSVM possesses within-class scatter and misclassification error properties.

Let the linear Pin-TBSVM seek two non-parallel hyperplanes of the form

$$f_1(\boldsymbol{x}) = \boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0 \text{ and } f_2(\boldsymbol{x}) = \boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0 \tag{4.1}$$

where $\boldsymbol{w}_1, \boldsymbol{w}_2 \in R^n$ and $b_1, b_2 \in R$ are unknowns. Replacing the hinge loss for the pinball loss function and the kernel empirical term in L_2 −norm, i.e., the second term of (2.27), by L_1 −norm, the linear Pin-TBSVM leads to solving the pair of optimization problems

$$\begin{gathered}\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2} \quad \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||\boldsymbol{\xi}_1||_1 + c_1 \boldsymbol{e}_2^t \boldsymbol{\xi}_2 \\ \text{subject to} \quad A\mathbf{w}_1 + b_1 \boldsymbol{e}_1 = \boldsymbol{\xi}_1, \\ -(B\boldsymbol{w}_1 + b_1 \boldsymbol{e}_2) \geq \boldsymbol{e}_2 - \boldsymbol{\xi}_2, \quad -(B\boldsymbol{w}_1 + b_1 \boldsymbol{e}_2) \leq \boldsymbol{e}_2 + \frac{1}{\tau_1} \boldsymbol{\xi}_2\end{gathered} \tag{4.2a}$$

and

$$\begin{gathered}\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||\boldsymbol{\eta}_2||_1 + c_2 \boldsymbol{e}_1^t \boldsymbol{\eta}_1 \\ \text{subject to} \quad B\mathbf{w}_2 + b_2 \boldsymbol{e}_2 = \boldsymbol{\eta}_2, \\ A\boldsymbol{w}_2 + b_2 \boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\eta}_1, \quad A\boldsymbol{w}_2 + b_2 \boldsymbol{e}_1 \leq \boldsymbol{e}_1 + \frac{1}{\tau_2} \boldsymbol{\eta}_1\end{gathered} \tag{4.2b}$$

where $c_k > 0$ and $c_{k+1} > 0$ ($k = 1, 2$) are regularization parameters; $\boldsymbol{\xi}_1, \boldsymbol{\eta}_1 \in R^{m_1}$; $\boldsymbol{\xi}_2, \boldsymbol{\eta}_2 \in R^{m_2}$ are vectors of slack variables and $0 \leq \tau_1, \tau_2 \leq 1$ are parameters.

Note that minimization of the terms $||\boldsymbol{\xi}_1||_1$ and $||\boldsymbol{\eta}_2||_1$will make the positive and negative class data points to be as close as possible to their corresponding hyperplanes. Problem (4.2) simultaneously considers the intra-class compactness and inter-class separation using absolute loss and pinball loss respectively.

Remark 4.1. When τ_1 and τ_2 are very small then the second inequality constraints in (4.2a) and (4.2b) will reduce to $\boldsymbol{\xi}_2 \geq \mathbf{0}$ and $\boldsymbol{\eta}_1 \geq \mathbf{0}$ respectively.

Remark 4.2. In problem (4.2), misclassification error is measured using pinball loss whereas LS loss is applied in L1-LSTBSVM.

Since $L_\tau(\boldsymbol{x}, y, f(\boldsymbol{x})) = \tau|(1 - yf(\boldsymbol{x}))| + (1 - \tau)L_{hinge}(\boldsymbol{x}, y, f(\boldsymbol{x}))$, i.e., the pinball loss is a linear combination of them, one may expect that in addition to noise insensitivity, Pin-TBSVM also possesses these properties.

Now we analyze the noise insensitivity, within-class scatter and misclassification error properties of Pin-TBSVM. For easy comprehension, we consider its linear formulation (4.2) in unconstrained form

$$\min_{\boldsymbol{w}_1, b_1} \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||A\boldsymbol{w}_1 + b_1 \boldsymbol{e}_1||_1 + c_1 \sum_{i=m_1+1}^{m_1+m_2} L_{\tau_1}(\boldsymbol{x}_i, y_i, f_1(\boldsymbol{x}_i)) \tag{4.3a}$$

and

$$\min_{\boldsymbol{w}_2, b_2} \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||B\boldsymbol{w}_2 + b_2 \boldsymbol{e}_2||_1 + c_2 \sum_{i=1}^{m_1} L_{\tau_2}(\boldsymbol{x}_i, y_i, f_2(\boldsymbol{x}_i)). \tag{4.3b}$$

Define a generalized sign function

$$sgn_{\tau}(u) = \begin{cases} 1, & u > 0 \\ [-\tau, 1], & u = 0 \\ -\tau & u < 0 \end{cases}$$

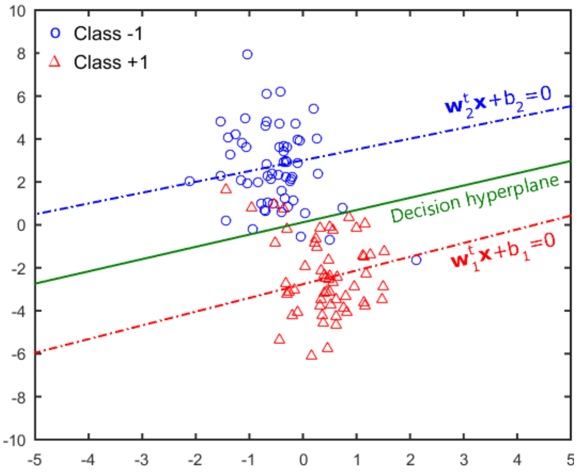
Then, using the optimality condition for (4.3a), we obtain

$$\mathbf{0} \in \frac{c_3}{c_1}\begin{bmatrix}\mathbf{w}_1 \\ b_1\end{bmatrix} + \frac{[A\ \mathbf{e}_1]^t}{c_1} sgn_{\tau=1}(\,[A\ \mathbf{e}_1]\begin{bmatrix}\mathbf{w}_1 \\ b_1\end{bmatrix}) + \sum_{i=m_1+1}^{m_1+m_2} sgn_{\tau_1}(1 + \mathbf{w}_1^t\mathbf{x}_i + b_1)\begin{bmatrix}\mathbf{x}_i \\ 1\end{bmatrix}. \tag{4.4}$$

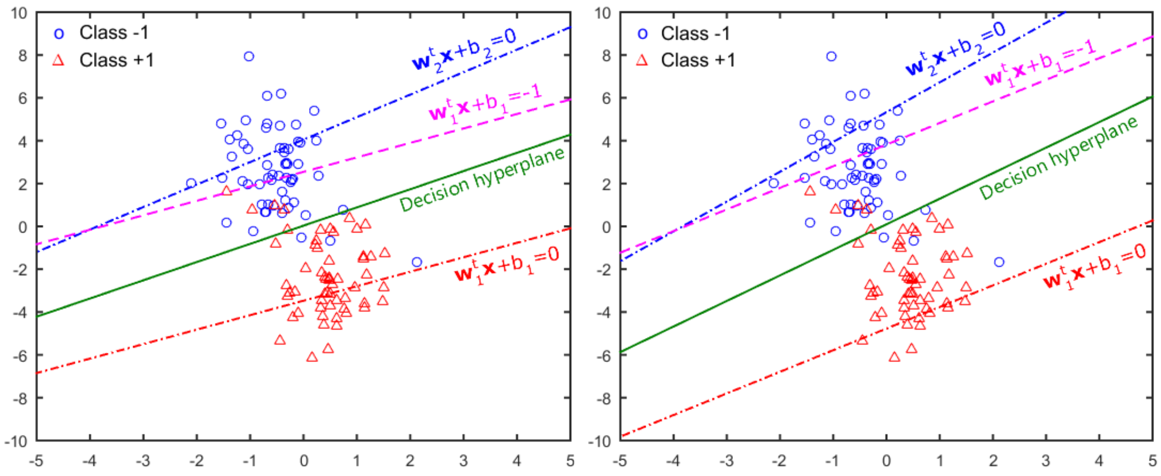
Let $S_1^+ = \{i{:}\, \mathbf{w}_1^t\mathbf{x}_i + b_1 + 1 \ > 0\}$, $S_1^- = \{i{:}\, \mathbf{w}_1^t\mathbf{x}_i + b_1 + 1 \ < 0\}$ and $S_1^0 = \{i{:}\, \mathbf{w}_1^t\mathbf{x}_i + b_1 + 1 \ = 0\}$ be the index sets where $i = m_1 + 1, \ldots, m_1 + m_2$. Then, the above condition (4.4) can be written as

$$\frac{c_3}{c_1}\begin{bmatrix}\mathbf{w}_1 \\ b_1\end{bmatrix} + \frac{[A\ \mathbf{e}_1]^t}{c_1} sgn_{\tau=1}(\,[A\ \mathbf{e}_1]\begin{bmatrix}\mathbf{w}_1 \\ b_1\end{bmatrix}) + \sum_{i\in S_1^+}\begin{bmatrix}\mathbf{x}_i \\ 1\end{bmatrix} - \tau_1\sum_{i\in S_1^-}\begin{bmatrix}\mathbf{x}_i \\ 1\end{bmatrix} + \sum_{i\in S_1^0}\varsigma_i\begin{bmatrix}\mathbf{x}_i \\ 1\end{bmatrix} = \mathbf{0} \tag{4.5}$$

where $\varsigma_i \in [-\tau_1, 1]$. From condition (4.5), we see that $\tau_1$controls the amount of vectors for training in $S_1^+$and S_1^-. For small values of τ_1, the set S_1^+contains few vectors and the result is sensitive. When $\tau_1 = 1$, both sets contain large number of vectors, and the result is less susceptible to noise on $\mathbf{x}_i$. This property is illustrated in Figure 4.1 on a 2-D synthetic dataset consisting of two classes of training points where those from the positive and negative classes are marked by red triangles and blue circles respectively. In addition, we have also shown the non-parallel hyperplanes $\mathbf{w}_1^t\mathbf{x} + b_1 = 0$ and $\mathbf{w}_1^t\mathbf{x} + b_1 = -1$ along with the boundary of the decision hyperplane obtained for the linear case. In fact, the results for TBSVM and L1-Pin-TBSVM ($\tau_1 = 0.1$ and $\tau_1 = 0.5$) are illustrated in Figure 4.1(a)-Figure 4.1(c) In Figure 4.1(b)-Figure 4.1(c), the set of points of S_1^+will be the points lying below the hyperplane $\mathbf{w}_1^t\mathbf{x} + b_1 = -1$ whereas all points lying above this hyperplane will be S_1^-. From



(a) TBSVM



(b) L1-Pin-TBSVM $(\tau = 0.1)$

(c) L1-Pin-TBSVM $(\tau = 0.5)$

Figure 4.1 Classification results of linear (a) TBSVM, (b) L1-Pin-TBSVM with $\tau = 0.1$, (c) L1-Pin-TBSVM

Figure 4.1(b) and Figure 4.1(c) with $\tau_1 = 0.1$ and $\tau_1 = 0.5$, with the increase of τ_1 the number of points in S_1^+ region is becoming larger and accordingly the summation of $\begin{bmatrix}\boldsymbol{x}_i\\1\end{bmatrix}$ term in (4.5) is insensitive to noise on $\boldsymbol{x}_i \in S_1^+$. A similar analysis holds for (4.3b).

Let $\boldsymbol{x}_0 \in T$ be such that $\boldsymbol{w}_1^t\boldsymbol{x}_0 + b_1 = 0$ holds. By using the sum of the absolute distance between $\boldsymbol{x}_0 \in T$and $\boldsymbol{x}_i$ vectors from the positive class in the projected space related to $\boldsymbol{w}_1$, the scatter of $\boldsymbol{x}_i \in T$ around $\boldsymbol{x}_0$ can be defined as $\sum_{i=1}^{m_1}|\boldsymbol{w}_1^t(\boldsymbol{x}_i - \boldsymbol{x}_0)| = \sum_{i=1}^{m_1}|\boldsymbol{w}_1^t\boldsymbol{x}_i + b_1|= ||A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1||_1$.

Consider the following problem formulation

$$\min_{\boldsymbol{w}_1,b_1} \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1||_1 + c_5||\boldsymbol{e}_2 + B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2||_1. \tag{4.6}$$

The first term is the regularization term and by minimizing it, the structural risk is minimized. The second term might be viewed as the minimization of the scatter loss around the positive hyperplane. Lastly, minimization of the third term makes the negative class vectors to be pushed to lie around the plane $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 + 1 = 0$.

By adding the hinge loss misclassification error term $c_7\sum_{i=m_1+1}^{m_1+m_2} max\{0,1 - y_i(\boldsymbol{w}_1^t x_i + b_1)\}$ into (4.6) and by choosing $c_1 = c_5 + c_7$ and $\tau_1 = \frac{c_5}{c_1}$, our proposed pinball SVM (4.2) is obtained, i.e., the pinball loss minimization of a class of vectors can be interpreted as pushing the vectors to lie around a parallel hyperplane and minimizing the misclassification error together.

A similar outcome can be obtained for (4.3b). Note that, we have $0 \le \tau_1, \tau_2 \le 1$.

Based on the discussion, one can see that Pin-TBSVM can be thought of as a model that puts emphasis on within-class scatter, pushing the class of vectors to lie around a parallel hyperplane and misclassification error minimization together.

Assume that the data is contaminated by outliers and feature noise. Our principal concern in this study is in obtaining a robust classifier by constructing an elegant problem formulation in a simple form like the classical SVM. In this regard, it is suggested to obtain the solutions of Pin-TBSVM by solving an equivalent pair of QPPs each having m number of unknowns only. With this objective, the Pin-TBSVM problem (4.2) is equivalently formulated into the following novel pinball TBSVM in L_1 −norm (L1-Pin-TBSVM) problem

$$\begin{aligned} \min_{\boldsymbol{w}_1,b_1,\boldsymbol{\xi}_1,\boldsymbol{\xi}_2} \quad & \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + \boldsymbol{e}_1^t\boldsymbol{\xi}_1 + c_1\boldsymbol{e}_2^t\boldsymbol{\xi}_2 \\ \text{subject to} \quad & A\mathbf{w}_1 + b_1\boldsymbol{e}_1 \leq \boldsymbol{\xi}_1, \quad -(A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1) \leq \boldsymbol{\xi}_1, \\ & -(B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) \geq \boldsymbol{e}_2 - \boldsymbol{\xi}_2, \quad -(B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) \leq \boldsymbol{e}_2 + \frac{1}{\tau_1}\boldsymbol{\xi}_2 \end{aligned} \tag{4.7a}$$

and

$$\begin{aligned} \min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}_1,\boldsymbol{\eta}_2} \quad & \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + \boldsymbol{e}_2^t\boldsymbol{\eta}_2 + c_2\boldsymbol{e}_1^t\boldsymbol{\eta}_1 \\ \text{subject to} \quad & B\mathbf{w}_2 + b_2\boldsymbol{e}_2 \leq \boldsymbol{\eta}_2, \quad -(B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \leq \boldsymbol{\eta}_2, \\ & A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \geq \boldsymbol{e}_1 - \boldsymbol{\eta}_1, \quad A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \leq \boldsymbol{e}_1 + \frac{1}{\tau_2}\boldsymbol{\eta}_1 \end{aligned} \tag{4.7b}$$

The formulation L1-Pin-TBSVM (4.7) can be expressed as

$$\begin{aligned} \min_{\boldsymbol{w}_1,b_1,\boldsymbol{\xi}_1,\boldsymbol{\xi}_2} \quad & \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + \boldsymbol{e}_1^t\boldsymbol{\xi}_1 + c_1\boldsymbol{e}_2^t\boldsymbol{\xi}_2 \\ \text{subject to} \quad & -\begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} \leq \begin{bmatrix} \boldsymbol{\xi}_1 \\ \boldsymbol{e}_2 + (1/\tau_1)\boldsymbol{\xi}_2 \end{bmatrix}, \quad \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} \leq \begin{bmatrix} \boldsymbol{\xi}_1 \\ -\boldsymbol{e}_2 + \boldsymbol{\xi}_2 \end{bmatrix} \end{aligned}$$

and

$$\begin{aligned} \min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}_1,\boldsymbol{\eta}_2} \quad & \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + \boldsymbol{e}_2^t\boldsymbol{\eta}_2 + c_2\boldsymbol{e}_1^t\boldsymbol{\eta}_1 \\ \text{subject to} - & \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_2 \\ b_2 \end{bmatrix} \leq \begin{bmatrix} -\boldsymbol{e}_1 + \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix}, \quad \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_2 \\ b_2 \end{bmatrix} \leq \begin{bmatrix} \boldsymbol{e}_1 + (1/\tau_2)\boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix} \end{aligned}$$

Or equivalently in matrix form

$$\begin{aligned} \min_{\boldsymbol{z}_1,\boldsymbol{\xi}} \quad & \frac{1}{2}c_3||\boldsymbol{z}_1||^2 + \boldsymbol{e}^tD_{11}\boldsymbol{\xi} \\ \text{subject to} \quad & -G\boldsymbol{z}_1 \leq \boldsymbol{r}_1 + D_{12}\boldsymbol{\xi}, \quad G\boldsymbol{z}_1 \leq -\boldsymbol{r}_1 + \boldsymbol{\xi} \end{aligned} \tag{4.8a}$$

and

$$\min_{\boldsymbol{z}_2,\boldsymbol{\eta}} \quad \frac{1}{2}c_4||\boldsymbol{z}_2||^2 + \boldsymbol{e}^tD_{22}\boldsymbol{\eta} \tag{4.8b}$$

$$\text{subject to} \quad -G\boldsymbol{z}_2 \le -\boldsymbol{r}_2 + \boldsymbol{\eta}, \quad G\boldsymbol{z}_2 \le \boldsymbol{r}_2 + D_{21}\boldsymbol{\eta}$$

where

$\boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{w}_k \\ b_k \end{bmatrix} \in R^{n+1}$, $G = \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix} = [C \quad \boldsymbol{e}]$, $\boldsymbol{r}_1 = \begin{bmatrix} \boldsymbol{0}_1 \\ \boldsymbol{e}_2 \end{bmatrix}$, $\boldsymbol{r}_2 = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{0}_2 \end{bmatrix}$, $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}_1 \\ \boldsymbol{\xi}_2 \end{bmatrix}$, $\boldsymbol{\eta} \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix}$, $\boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{bmatrix}$; $\boldsymbol{0}_k$,$\boldsymbol{e}_k$ signify the zeros in the column vectors and ones of dimension m_k for $k = 1{,}2$: $D_{11} = diag(\boldsymbol{e}_1; c_1\boldsymbol{e}_2)$, $D_{12} = diag(\boldsymbol{e}_1; (1/\tau_1)\boldsymbol{e}_2)$, $D_{22} = diag(c_2\boldsymbol{e}_1; \boldsymbol{e}_2)$, $D_{21} = diag((1/\tau_2)\boldsymbol{e}_1; \boldsymbol{e}_2)$ are diagonal matrices of order *m*.

We solve the problem (4.8) in the dual space. By introducing the vectors of Lagrangian multipliers $\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2 \ge \boldsymbol{0}$ from R^m, the Lagrangian functions for (4.8) can be obtained as

$$\tilde{L}_1(\boldsymbol{z}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}_1, \boldsymbol{\beta}_1) = \frac{1}{2} c_3 \boldsymbol{z}_1^t \boldsymbol{z}_1 + \boldsymbol{e}^t D_{11} \boldsymbol{\xi} + \boldsymbol{\alpha}_1^t(-G\boldsymbol{z}_1 - \boldsymbol{r}_1 - D_{12}\boldsymbol{\xi}) + \boldsymbol{\beta}_1^t(G\boldsymbol{z}_1 + \boldsymbol{r}_1 - \boldsymbol{\xi})$$

and

$$\tilde{L}_2(\boldsymbol{z}_2, \boldsymbol{\eta}, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2) = \frac{1}{2} c_4 \boldsymbol{z}_2^t \boldsymbol{z}_2 + \boldsymbol{e}^t D_{22} \boldsymbol{\eta} + \boldsymbol{\alpha}_2^t(-G\boldsymbol{z}_2 + \boldsymbol{r}_2 - \boldsymbol{\eta}) + \boldsymbol{\beta}_2^t(G\boldsymbol{z}_2 - \boldsymbol{r}_2 - D_{21}\boldsymbol{\eta})$$

By applying the necessary and sufficient KKT conditions for optimality, we get

$$\partial\tilde{L}_k/\partial\boldsymbol{z}_k = \boldsymbol{0} \Rightarrow \boldsymbol{z}_k = \frac{G^t}{c_{k+2}}(\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k) \quad \text{for } k = 1, 2, \tag{4.9}$$

$$\partial\tilde{L}_1/\partial\boldsymbol{\xi} = \boldsymbol{0} \Rightarrow D_{12}\boldsymbol{\alpha}_1 + \boldsymbol{\beta}_1 = D_{11}\boldsymbol{e}, \quad \partial\tilde{L}_2/\partial\boldsymbol{\eta} = \boldsymbol{0} \Rightarrow \boldsymbol{\alpha}_2 + D_{21}\boldsymbol{\beta}_2 = D_{22}\boldsymbol{e}, \tag{4.10}$$

$$\boldsymbol{\alpha}_1^t(-G\boldsymbol{z}_1 - \boldsymbol{r}_1 - D_{12}\boldsymbol{\xi}) = 0, \boldsymbol{\beta}_1^t(G\boldsymbol{z}_1 + \boldsymbol{r}_1 - \boldsymbol{\xi}) = 0, \tag{4.11}$$

$$\boldsymbol{\alpha}_2^t(-G\boldsymbol{z}_2 + \boldsymbol{r}_2 - \boldsymbol{\eta}) = 0, \boldsymbol{\beta}_2^t(G\boldsymbol{z}_2 - \boldsymbol{r}_2 - D_{21}\boldsymbol{\eta}) = 0 \text{ and } \boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2 \ge \boldsymbol{0}. \tag{4.12}$$

Using (4.9) and (4.10), the duals of (4.8) can be obtained as a pair of QPP minimization problems

$$\begin{gathered} \min_{\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1 \in R^m} \frac{1}{2c_3}(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1)^t GG^t(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1) + \boldsymbol{r}_1^t(\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1) \\ \text{subject to } D_{12}\boldsymbol{\alpha}_1 + \boldsymbol{\beta}_1 = D_{11}\boldsymbol{e} \text{ and } \boldsymbol{\alpha}_1 \ge \boldsymbol{0}, \boldsymbol{\beta}_1 \ge \boldsymbol{0} \end{gathered} \tag{4.13a}$$

and

$$\begin{gathered} \min_{\boldsymbol{\alpha}_2, \boldsymbol{\beta}_2 \in R^m} \frac{1}{2c_4}(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2)^t GG^t(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2) - \boldsymbol{r}_2^t(\boldsymbol{\alpha}_2 - \boldsymbol{\beta}_2) \\ \text{subject to } \boldsymbol{\alpha}_2 + D_{21}\boldsymbol{\beta}_2 = D_{22}\boldsymbol{e} \text{ and } \boldsymbol{\alpha}_2 \ge \boldsymbol{0}, \boldsymbol{\beta}_2 \ge \boldsymbol{0}. \end{gathered} \tag{4.13b}$$

Using the condition $D_{12}\boldsymbol{\alpha}_1 + \boldsymbol{\beta}_1 = D_{11}\boldsymbol{e}$ we get $\boldsymbol{\alpha}_1 - \boldsymbol{\beta}_1 = (I + D_{12})\boldsymbol{\alpha}_1 - D_{11}\boldsymbol{e}$ and $\boldsymbol{\beta}_1 = D_{11}\boldsymbol{e} - D_{12}\boldsymbol{\alpha}_1 \ge \boldsymbol{0}$. Putting this results in the dual problem (4.13a), we obtain

$$\min_{\boldsymbol{\alpha}_1 \in R^m} L_1(\boldsymbol{\alpha}_1) = \frac{1}{2}\boldsymbol{\alpha}_1^t Q_1 \boldsymbol{\alpha}_1 - \boldsymbol{h}_1^t \boldsymbol{\alpha}_1, \tag{4.14a}$$

$$\text{subject to } \mathbf{0} \leq \boldsymbol{\alpha}_1 \leq D_{12}^{-1}D_{11}\boldsymbol{e} = \begin{pmatrix} \boldsymbol{e}_1 \\ c_1\tau_1\boldsymbol{e}_2 \end{pmatrix}$$

where $Q_1 = (I + D_{12})GG^t(I + D_{12})\ and\ \boldsymbol{h}_1 = (I + D_{12})(GG^tD_{11}\boldsymbol{e} - c_3\boldsymbol{r}_1)$.

Again, since $\boldsymbol{\alpha}_2 + D_{21}\boldsymbol{\beta}_2 = D_{22}\boldsymbol{e}$ and therefore $\boldsymbol{\alpha_2} - \boldsymbol{\beta}_2 = D_{22}\boldsymbol{e} - (I + D_{21})\boldsymbol{\beta}_2$ implies problem (4.13b) can be rewritten in simple form as

$$\min_{\boldsymbol{\beta}_2\in R^m} L_2(\boldsymbol{\beta}_2) = \frac{1}{2}\boldsymbol{\beta}_2^tQ_2\boldsymbol{\beta}_2 - \boldsymbol{h}_2^t\boldsymbol{\beta}_2, \text{subject to } \mathbf{0} \leq \boldsymbol{\beta}_2 \leq D_{21}^{-1}D_2\boldsymbol{e} = \begin{pmatrix} c_2\tau_2\boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix} \tag{4.14b}$$

where $Q_2 = (I + D_{21})GG^t(I + D_{21})\ and\ \boldsymbol{h}_2 = (I + D_{21})(GG^tD_{22}\boldsymbol{e} - c_4\boldsymbol{r}_2)$.

Using the solutions of (4.14) and (4.9), the decision function for L1-Pin-TBSVM is obtained

$$f(\boldsymbol{x}) = sign\left(\frac{\boldsymbol{w}_1^t\boldsymbol{x} + b_1}{||\boldsymbol{w}_1||} + \frac{\boldsymbol{w}_2^t\boldsymbol{x} + b_2}{||\boldsymbol{w}_2||}\right).$$

Remark 4.3. In the derivation of L1-Pin-TBSVM (4.14), it is assumed that $(\tau_1, \tau_2) \neq (0,0)$. Notice that our proposed method of solving (4.14) is novel and is general in the sense that it is applicable for $\tau_1, \tau_2 \in (0,1]$.

Remark 4.4. Problems (4.14a) and (4.14b) are convex QPPs with box constraints in simple form. Since L1-Pin-TBSVM solves the pair of QPPs (4.14) in m variables, its computational complexity for learning is $2O(m^3)$.

Remark 4.5. When $\tau_1 = \tau_2 = 0$, problem (4.2) is the same as the L_1 −norm loss based twin SVM (L1LTSVM) considered for study by Peng et al. (2016).

Remark 4.6. Unlike in TBSVM where the dual QPPs of (2.27) contain terms having the inverse of kernel matrices (Shao et al., 2011), the proposed L1-Pin-TBSVM has the advantage that its dual QPPs (4.14) are free of matrix inversion terms.

Remark 4.7. From the constraints of the problem (4.14) and from (4.10)-(4.12), one can verify that $\boldsymbol{\alpha}_1, \boldsymbol{\beta}_2$ become sparse and hence from (4.9) we say that the normal vectors $\boldsymbol{w}_k$where $k = 1,2$ are partly sparse (Peng et al., 2016).

Remark 4.8. The pair of dual problems (4.14) can be solved efficiently using SVM type algorithms like sequential minimal optimization (SMO) (Platt, 1999), dual coordinate descent (DCD) (Hsieh et al., 2008), successive over-relaxation (SOR) (Mangasarian & Wild, 2006), clipping DCD (clipDCD) (Peng et al., 2014). However, in this work, MOSEK optimization toolbox available for MATLAB is used.

For the non-linear case, we proceed as follows. Assuming that $\varphi(.)$is the non-linear mapping taking the training vectors into a higher dimensional feature space, Pin-TBSVM in the feature space can be formulated in L_1 −norm as

$$\begin{aligned} \min_{\boldsymbol{w}_1, b_1, \xi_1, \xi_2} \quad & \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||\varphi(A)\boldsymbol{w}_1 + b_1\boldsymbol{e}_1||_1 + c_1\boldsymbol{e}_2^t\boldsymbol{\xi}_2 \\ & \text{subject to} - (\varphi(B)\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) \geq \boldsymbol{e}_2 - \boldsymbol{\xi}_2, \end{aligned} \tag{4.15a}$$

$$-(\varphi(B)\boldsymbol{w}_1 + b_1\boldsymbol{e}_2) \le \boldsymbol{e}_2 + \frac{1}{\tau_1}\boldsymbol{\xi}_2$$

and

$$\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||\varphi(B)\boldsymbol{w}_2 + b_2\boldsymbol{e}_2||_1 + c_2\boldsymbol{e}_1^t\boldsymbol{\eta}_1$$

$$\text{subject to} \quad \varphi(A)\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \ge \boldsymbol{e}_1 - \boldsymbol{\eta}_1, \quad \varphi(A)\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 \le \boldsymbol{e}_1 + \frac{1}{\tau_2}\boldsymbol{\eta}_1. \qquad (4.15b)$$

By taking $G = \begin{bmatrix} \varphi(A) & \boldsymbol{e}_1 \\ \varphi(B) & \boldsymbol{e}_2 \end{bmatrix} = [\varphi(C) \quad \boldsymbol{e}]$, therefore $GG^t = K(C, C^t) + E$ where $E = \boldsymbol{e}\boldsymbol{e}^t$is the matrix in which all its entries are equal to 1, and proceeding as in the linear case, the dual problems of (4.15) are obtained to be a pair of QPPs of the form (4.14). In this case,

$$\boldsymbol{z}_k = [\varphi(C)\boldsymbol{e}]^t\boldsymbol{u}_k \text{ and } f_k(\boldsymbol{x}) = [\varphi(\boldsymbol{x})1]\begin{bmatrix}\boldsymbol{w}_k \\ b_k\end{bmatrix} = (K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_k \text{ for } k = 1{,}2,$$

where $\boldsymbol{u}_1 = ((I + D_{12})\boldsymbol{\alpha}_1 - D_{11}\boldsymbol{e})/c_3$ and $\boldsymbol{u}_2 = (D_{22}\boldsymbol{e} - (I + D_{21})\boldsymbol{\beta}_2)/c_3$.

Finally, the decision function for non-linear L1-Pin-TBSVM can be obtained as

$$f(\boldsymbol{x}) = sign\left(\frac{(K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_1}{||\boldsymbol{u}_1^t K(C, C^t)\boldsymbol{u}_1||} + \frac{(K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_2}{||\boldsymbol{u}_2^t K(C, C^t)\boldsymbol{u}_2||}\right)$$

Notice that when $K(C, C^t) = CC^t$, problem (4.14) for the non-linear case is reduced to a linear problem, i.e. the proposed L1-Pin-TBSVM allows a uniform formulation for the linear and non-linear kernels.

4.3 Discussion on L1-Pin-TBSVM

we compare the proposed L1-Pin-TBSVM to TBSVM, Pin-SVM and Pin-TSVM in this subsection.

4.3.1 LI-Pin-TBSVM versus TBSVM

Both TBSVM and L1-Pin-TBSVM classifiers are based on non-parallel hyperplanes and they lead to solving a pair of QPPs with box constraints. TBSVM uses the hinge loss whereas in L1-Pin-TBSVM, pinball loss is used. In case of non-linear classification, TBSVM identifies kernel generated surfaces of the kind (2.26), which means that an approximate formulation is solved (Tian, 2013). On the contrary, non-linear L1-Pin-TBSVM is formulated by applying the kernel trick directly in the dual QPPs like in the classical SVM and thereby a unified form for the linear and non-linear kernel is obtained. Although a pair of smaller sized QPPs (2.27) is solved in TBSVM, i.e. if the sizes of the datasets for the two classes are approximately equal then its computational training complexity is $2O((m/2)^3)$, its formulation requires the inverse of kernel matrices of additional computational complexity $2O((m+1)^3)$ which in practice is intractable. However, no inverse matrices appear in L1-Pin-TBSVM. Compared to the minimization of the empirical kernel term of TBSVM in L_2 −norm, the kernel term in L1-Pin-TBSVM as the scatter loss is minimized in L_1 −norm and hence the proposed L1-Pin-TBSVM is robust to feature noise and outliers.

4.3.2 LI-Pin-TBSVM versus Pin-SVM

Since pinball loss function is used in L1-Pin-TBSVM and therefore, like Pin-SVM, the proposed L1-Pin-TBSVM has both noise insensitivity and re-sampling stability properties. With *(m+1)* equality constraints Pin-SVM resolves a single QPP in *2m* nonnegative variables and hence its computational training complexity is $O((2m)^3)$ whereas the computational complexity of L1-Pin-TBSVM is $2O(m^3)$. This makes L1-Pin-TBSVM a faster learning method than Pin-SVM.

4.3.3 LI-Pin-TBSVM versus Pin-TSVM

Though the problem formulations of L1-Pin-TBSVM and Pin-TSVM are entirely different, both the methods share certain important properties: (i). pinball loss is used; (ii). a pair of QPPs is solved. In fact, when the sizes of the datasets of both the classes are approximately equal, i.e. $m_1 \approx m_2 \approx m/2$, Pin-TSVM solves a pair of QPPs having m nonnegative variables and $(m/2+1)$ equality constraints resulting its computational complexity is $2O(m^3)$, i.e. same order of complexity is obtained for both the methods; (iii). The kernel technique is directly used and non-linear formulation is obtained; (iv). formulations are free from inverse of kernel matrices; (v). noise insensitivity and small within-class scatter properties are satisfied.

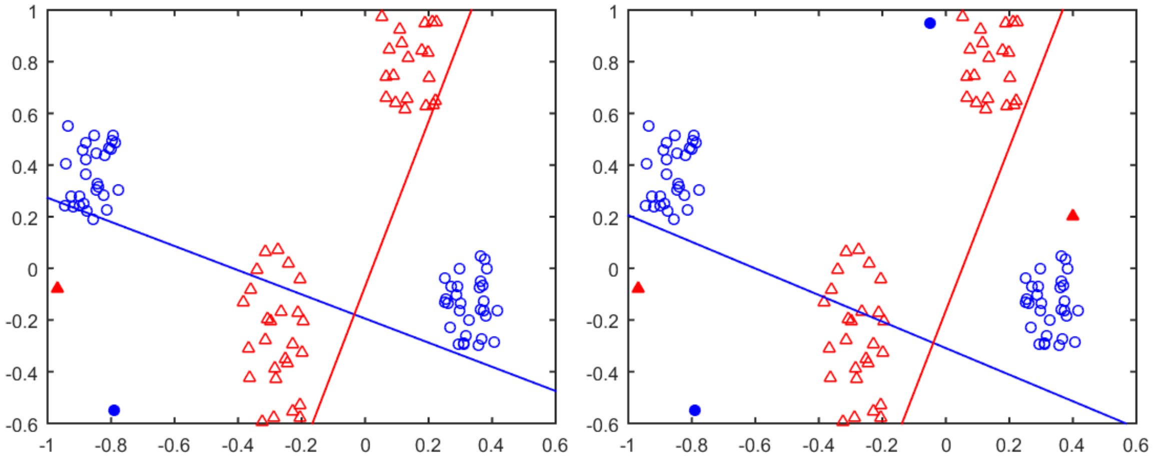
4.4 Experimental Results

To investigate the effectiveness of the proposed L1-Pin-TBSVM, we perform an empirical comparison between L1-Pin-TBSVM and, along with the classical SVM, few popular variants: TBSVM, L1-LSTBSVM, Pin-SVM and Pin-TSVM. All the experiments are performed in MATLAB R2015a environment on a PC running Windows 10 operating system with 64 bit Intel®core™ i7 CPU with 8 GB of RAM. For solving the QPPs, MOSEK optimization toolbox for MATLAB available at http://www.mosek.com is used. In experiments, the popular Gaussian kernel function of the form: $k(\boldsymbol{x},\boldsymbol{z}) = exp(-||\boldsymbol{x}-\boldsymbol{z}||^2/2\sigma^2)$ is used where $\sigma > 0 > 0$ is the parameter. The parameters for SVMs are chosen by tenfold cross-validation strategy where the penalty parameters c's, ν_1, ν_2 and the kernel parameter σ are selected from the set of values $\{2^i|$ i = -9, -8,-7,…,9} whereas τ_1, τ_2 are chosen from {0.1, 0.2, 0.5, 1}, i.e., (i). by randomly splitting each dataset into ten folds and using one of them as the validation set and the remaining for training, the accuracy over the validation set is computed; (ii). the values of the parameters showing the highest average accuracy are chosen as their optimal values where the average accuracy is computed by considering each one of the remaining folds as a validation set. For brevity's sake, we assume $c_1 = c_2,\ c_3 = c_4,\ v_1 = v_2$ and $\tau = \tau_1 = \tau_2$.

To validate the effectiveness of L1-Pin-TBSVM in the presence of outliers and feature noise, the results on *Crossplanes* (Mangasarian & Wild, 2006) and two-moon synthetic datasets are described. For empirical comparisons of the algorithms, 11 benchmark datasets from the UCI repository of machine learning datasets (Murphy & Aha, 1992) are considered for noiseless and noisy settings.

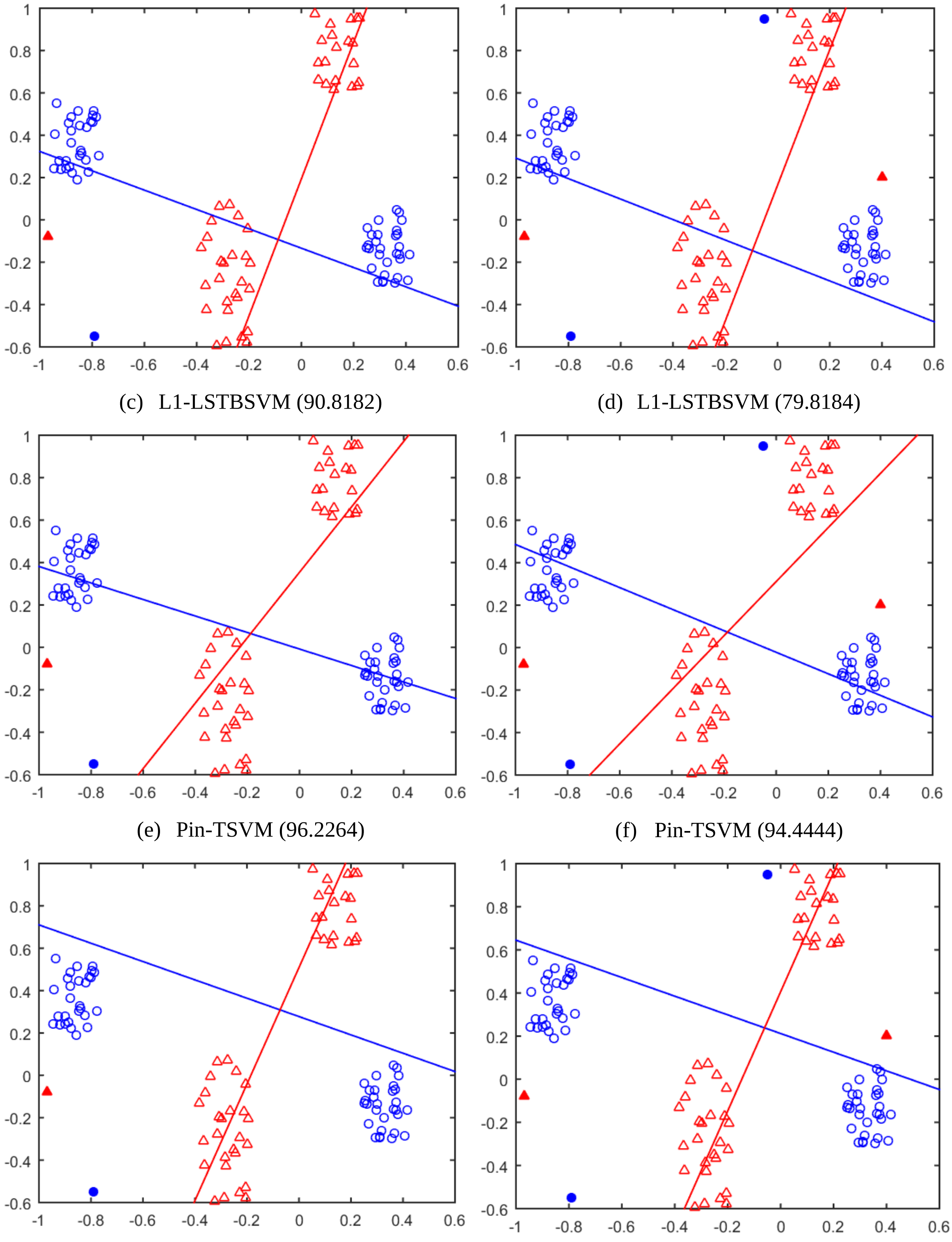
4.4.1 Crossplanes synthetic dataset

To verify the classification performance of L1-Pin-TBSVM in the presence of outliers, a two dimensional *Crossplanes* dataset consisting of 46 positive and 58 negative classes of points is considered. It is generated by perturbing points originally lying on two intersecting lines and belonging in two classes. In Figure 4.2, the positive and negative points of *Crossplanes* are shown by using red and blue colours filled triangles and circles respectively. As it is known that the presence of outliers will degrade the performance of any classification method, for the purpose of testing robustness, manually we added outliers two in number in one case and four in the other case. They are marked by filled triangles and circles in red and blue colours in Figure 4.2. We learned the non-parallel optimal hyperplanes on *Crossplanes* in the presence of two and four outliers, and their corresponding linear results for TBSVM, L1-LSTBSVM, Pin-TSVM and L1-Pin-TBSVM are shown in Figure 4.2. For the problem of two outliers, the results of classification accuracies of L1-Pin-TBSVM, Pin-TSVM, L1-LSTBSVM and TBSVM are 98.1%, 96.2%, 90.8% and 81.1% respectively whereas for the case of four outliers, they are 96.3%, 94.4%, 79.8% and 66.7% respectively. This clearly shows the influence of outliers on the performance of a learning



(a) TBSVM (81.1321)

(b) TBSVM (66.6667)



(c) L1-LSTBSVM (90.8182)

(d) L1-LSTBSVM (79.8184)

(e) Pin-TSVM (96.2264)

(f) Pin-TSVM (94.4444)

(g) L1-Pin-TBSVM (98.1132)

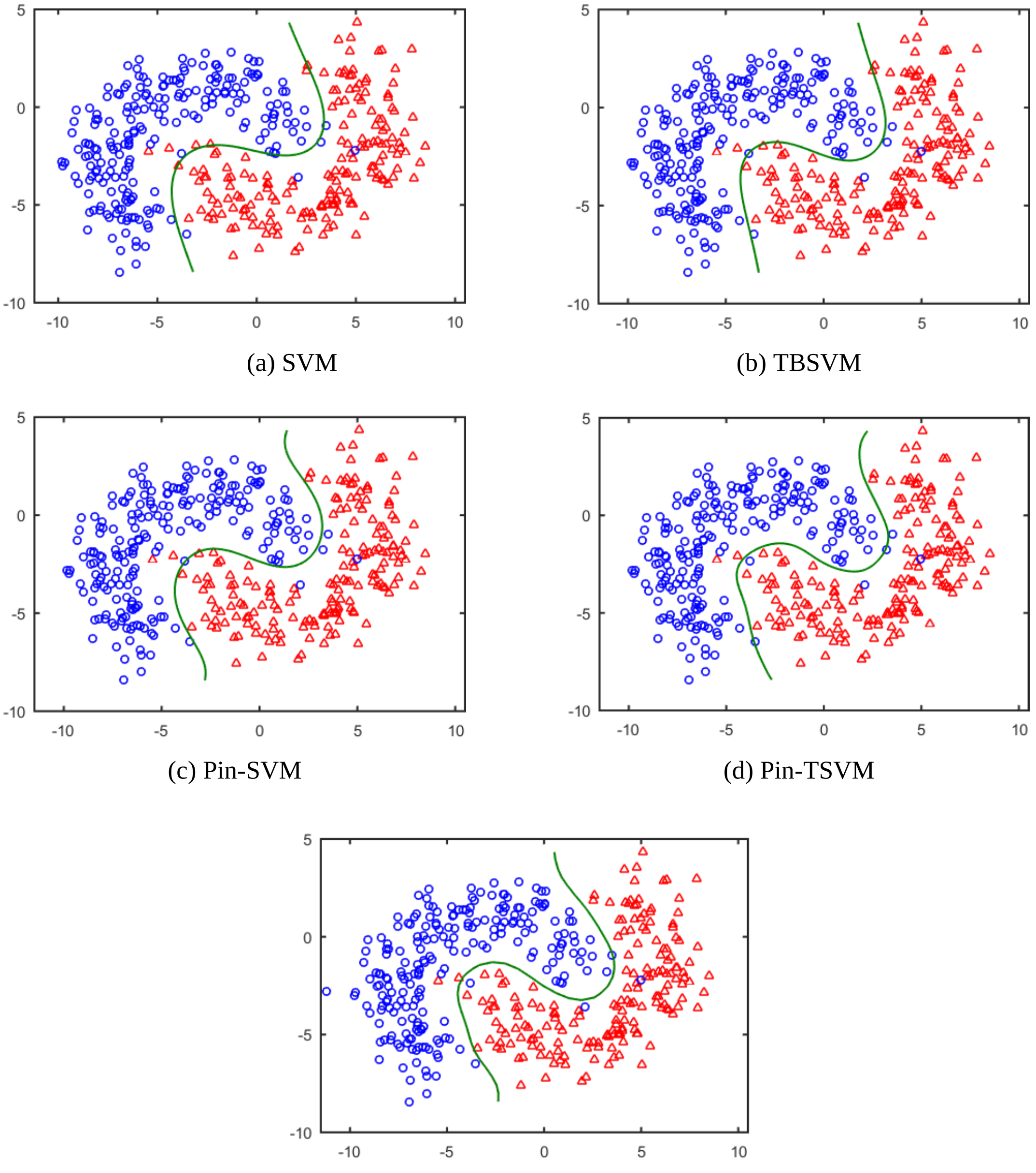
(h) L1-Pin-TBSVM (96.2963)

Figure 4.2 Results of nonparallel hyperplanes generated by linear TBSVM, L1-LSTBSVM, Pin-TSVM and L1-Pin-TBSVM on “crossplanes” dataset with two outliers (Left) and four outliers (Right). ‘Triangles’ and ‘circles’ represent positive and negative class points. Outliers are marked by filled triangles and circles.

method. Among the methods considered, better performance and very small performance degradation from 98.1% to 96.3% of L1-Pin-TBSVM show its superiority in terms of efficiency and robustness to outliers. The superior performance of L1-Pin-TBSVM in comparison to TBSVM is attributed to L_1 −norm is known for its ability to suppress the negative effect of outliers whereas the worst performance degradation from 81.1% to 66.7% of TBSVM where L_2 −norm is applied clearly shows its sensitivity towards outliers. Though L_1 −norm is less sensitive to outliers, the poor performance of L1-LSTBSVM in comparison to L1-Pin-TBSVM may be because of the non-robust LS loss function. From this discussion, we conclude that L1-Pin-TBSVM is a powerful method for data classification having the ability to suppress the influence of outliers.

4.4.2 Two-moon synthetic dataset

To illustrate the effectiveness and robustness of L1-Pin-TBSVM, we considered a two-dimensional synthetic binary classification dataset having two-moon shape shown in Figure 4.3. It consists of 220 positive and 180 negative samples where they are represented by small red and blue triangles and circles, respectively. As seen from the figure, it is linearly non-separable. To validate the insensitivity of L1-Pin-TBSVM towards noise, each feature of the data is corrupted by zero-mean additive Gaussian noise and its standard deviation r is chosen as 1, 1.3 and 1.5. One-run simulation results showing the classifiers of the non-linear SVM, TBSVM, Pin-SVM, Pin-TSVM and L1-Pin-TBSVM for $r = 1$and $r = 1.5$ are illustrated graphically in Figure 4.3(a) and Figure 4.3(b) respectively. By repeating the process of randomly generating 400 test vectors and computing the test accuracies, its average test accuracy is reported in Table 4.1. The classification accuracies of L1-Pin-TBSVM and its comparisons for Gaussian kernel are summarized for $r = 1, r = 1.3$ and $r = 1.5$. Here, higher accuracy indicates better performance. The highest classification accuracy is shown in boldface and it is achieved by L1-Pin-TBSVM at all levels of noise. As the level of noise r increases, performance degradation by all the learning methods shows their sensitivity to noise. Although L1-Pin-TBSVM and the hinge loss based SVMs show nearly equal accuracy for $r = 1$, the results of comparison of the performance degradations for $r = 1.5$ demonstrates that L1-Pin-TBSVM is more robust to SVM and TBSVM in the presence of noise. Similar result holds for Pin-SVM and Pin-TSVM. This shows that pinball loss is more robust to feature noise in comparison to hinge loss and better classification performance of L1-Pin-TBSVM confirms its superiority to all learning methods considered.



(a) SVM

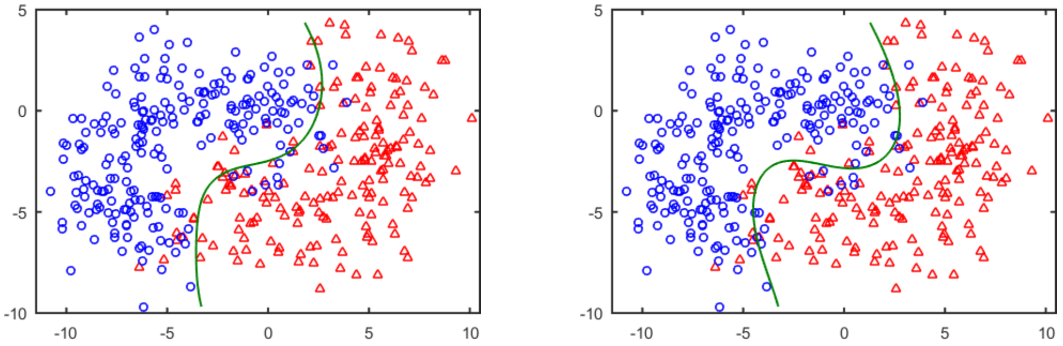
(b) TBSVM

(c) Pin-SVM

(d) Pin-TSVM

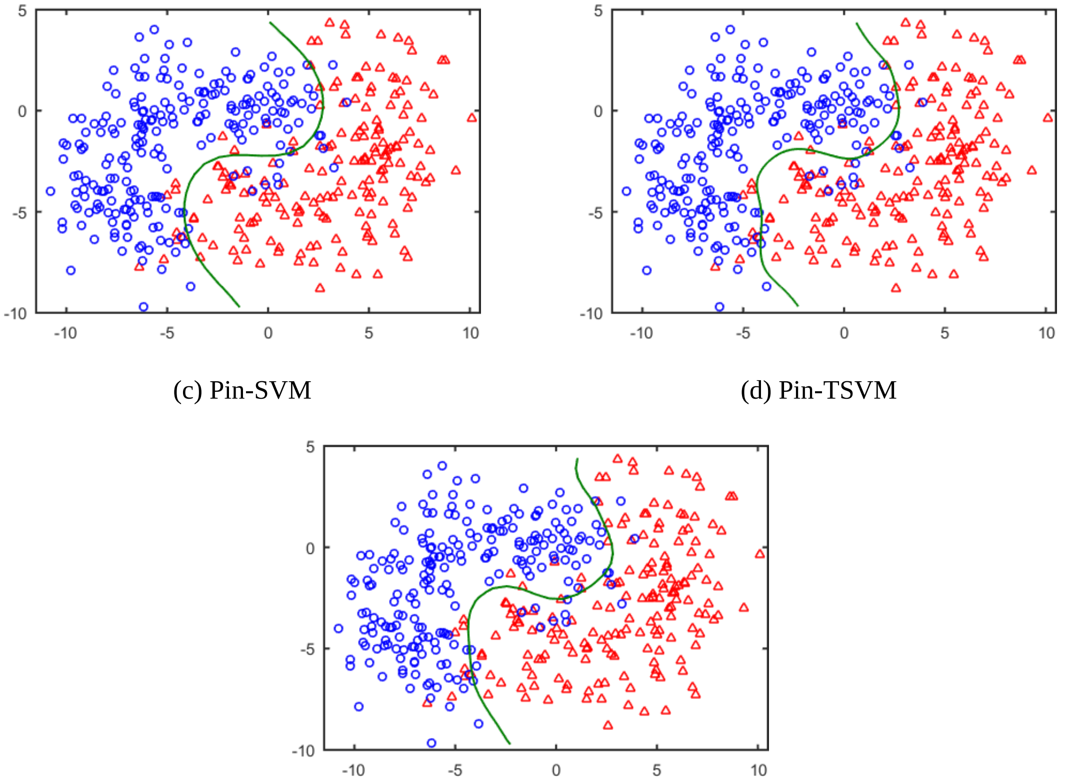
(e) L1-Pin-TBSVM

Figure 4.3(a) Visualizations of two-moon dataset and classifiers trained by SVM, TBSVM, Pin-SVM, Pin-TSVM, L1-Pin-TBSVM with noise level *r*=1.



(a) SVM

(b) TBSVM



(c) Pin-SVM

(d) Pin-TSVM

(e) L1-Pin-TBSVM

Figure 4.4(b) Visualization of two-moon dataset and classifiers trained by SVM, TBSVM, Pin-SVM, Pin-TSVM, L1-Pin-TBSVM with noise level *r*=1.5.

Table 4.1 A comparison of the proposed method's performance L1-Pin-TBSVM with SVM, TBSVM, Pin-SVM and Pin-TSVM on two-moon dataset with different levels of noise. The Gaussian kernel was used. Best result is indicated in boldface.

Dataset (Total size)	Level of noise	**SVM**	**TBSVM**	**Pin-SVM**	**Pin-TSVM**	**L1-Pin-TBSVM**
Two-moon (400 X 2)	*r*=1	98.12±0.6553	98.45±5.6090	98.55±0.5765	98.83±0.5288	**98.93±0.5280**
	r=1.3	95.81±0.8945	95.94±0.9965	96.45±0.7406	96.47±0.7408	**97.05±0.6181**
	r=1.5	92.77±1.0542	92.91±1.0160	93.92±0.9692	93.98±0.9697	**94.31±0.9358**

4.4.3 **Benchmark datasets**

In this subsection, further, we test the effectiveness of the proposed method for linear and Gaussian kernels in terms of classification accuracy and learning time by performing experiments on 11 popular benchmark datasets of UCI (Murphy & Aha, 1992) under noiseless and noisy settings. All the samples are normalized so that their feature values lie in the unit interval [0, 1]. As noise setting, each feature value is corrupted by zero-mean Gaussian noise where the ratio (r) of variance of noise to that of feature is taken as $r = 0$ (i.e., noise-free), $r = 0.05$ and $r = 0.1$ (Huang et al., 2014). Both the training and test vectors are corrupted by the same noise. In Table 4.2 and Table 4.4, we report the experimental results for linear and Gaussian kernels including the learning CPU time, the average cross-validation accuracy as the classification accuracy and the standard deviation. We ranked the algorithms in accordance with the classification accuracy obtained on each dataset and is reported.

Table 4.2 A comparison of the performance of proposed L1-Pin-TBSVM with SVM, TBSVM, Pin-SVM, Pin-TSVM on benchmark datasets with noise level r. The linear kernel was used. Best result is indicated in boldface.

Dataset (Total size)	Level of noise	SVM Time (C) Rank	TBSVM Time $(C1, C2)$ Rank	Pin-SVM Time $(C1, \tau)$ Rank	Pin-TSVM Time $(C1, \tau)$ Rank	L1-Pin-TBSVM Time $(C1, C2, \tau)$ Rank
Sonar (208×60)	*r*=0	78.38±5.97 0.0047 (2^{-2}) (4)	78.95±5.82 0.0024 (2^{0}, 2^{-1}) (2)	78.14±5.75 0.0176 (2^{4}, 0.5) (5)	78.52±5.92 0.0104 (2^{2}, 0.1) (3)	**79.57±5.87** 0.0018 (2^{-2}, 2^{4},1) (1)
	r=0.05	77.47±6.16 0.0082 (2^{-3}) (5)	77.52±5.92 0.0031 (2^{1}, 2^{0}) (4)	78.35±6.26 0.0130 (2^{3}, 0.2) (2)	77.57±5.69 0.0128 (2^{4}, 0.8) (3)	**78.96±5.56** 0.0045 (2^{-2}, 2^{4},0.8) (1)
	r=0.1	76.42±6.27 0.0074 (2^{-2}) (5)	77.34±4.28 0.0045 (2^{4}, 2^{4}) (3)	78.05±5.86 0.0144 (2^{3}, 0.8) (2)	77.14±6.52 0.0165 (2^{4}, 1) (4)	**78.44±5.74** 0.0060 (2^{-3}, 2^{4},0.8) (1)
Breast-cancer (683×9)	*r*=0	95.79±2.96 0.0340 (2^{-1}) (4)	95.06±4.37 0.0148 (2^{3}, 2^{4}) (5)	96.05±3.66 0.0413 (2^{-4}, 0.1) (2)	95.93±2.31 0.0225 (2^{4}, 1) (3)	**96.44±3.21** 0.0185 (2^{2}, 2^{-2}, 2^{0},1) (1)
	r=0.05	95.37±2.72 0.0334 (2^{0}) (4)	94.64±2.73 0.0138 (2^{-2}, 2^{4}) (5)	95.87±3.48 0.0422 (2^{-3}, 0.5) (2)	95.58±2.46 0.0239 (2^{4}, 0.8) (3)	**96.02±2.36** 0.0160 (2^{2},2^{-2},0.5) (1)
	r=0.1	94.50±3.29	94.39±2.60	94.95±3.52	94.71±4.11	**95.40±4.18**

		0.0392 (2^{-4}) (4)	0.0110 $(2^{0}, 2^{4})$ (5)	0.0448 $(2^{-3}, 0.8)$ (2)	0.0281 $(2^{0}, 0.2)$ (3)	0.0163 $(2^{2},2^{-2},0.2)$ (1)
German (1000×24)	r=0	75.50±6.05 0.0734 (2^{3}) (5)	75.70±6.83 0.0319 $(2^{3}, 2^{1})$ (3.5)	75.70±5.89 0.0843 $(2^{-4}, 0.1)$ (3.5)	75.81±5.94 0.0440 $(2^{-3}, 1)$ (2)	**76.90±5.15** 0.0477 $(2^{4},2^{-4},0.1)$ (1)
	r=0.05	74.80±5.39 0.0716 (2^{3}) (3)	74.20±4.56 0.0373 $(2^{-4}, 2^{1})$ (5)	74.25±4.38 0.0865 $(2^{-2}, 0.5)$ (4)	75.30±5.41 0.0469 $(2^{0}, 0.2)$ (2)	**76.50±4.24** 0.0482 $(2^{4},2^{-3},0.2)$ (1)
	r=0.1	74.30±4.39 0.0783 (2^{0}) (2)	73.00±5.55 0.0325 $(2^{-2}, 2^{2})$ (5)	73.40±5.91 0.0883 $(2^{-3}, 0.2)$ (4)	74.00±4.80 0.0498 $(2^{-1}, 0.8)$ (3)	**75.40±4.37** 0.0494 $(2^{3},2^{-3},0.5)$ (1)
CMC (1473 X 9)	r=0	**74.95±4.93** 0.0935 (2^{-4}) (1.5)	73.95±4.93 0.0585 $(2^{-4}, 2^{-1})$ (4.5)	73.95±4.93 0.1439 $(2^{-4}, 0.1)$ (4.5)	74.02±4.98 0.0672 $(2^{-2}, 0.5)$ (3)	**74.95±3.54** 0.0533 $(2^{1},2^{-1},0.2)$ (1.5)
	r=0.05	73.58±4.48 0.0988 (2^{-4}) (4)	73.76±4.93 0.0592 $(2^{-3}, 2^{-1})$ (3)	73.54±5.03 0.1466 $(2^{-4}, 0.8)$ (5)	**73.95±4.93** 0.0643 $(2^{-2}, 0.8)$ (1.5)	**73.95±3.24** 0.0546 $(2^{1},2^{-2},1)$ (1.5)
	r=0.1	73.02±4.35 0.0993 (2^{-4}) (5)	73.31±4.82 0.0552 $(2^{-4}, 2^{-1})$ (3)	73.11±5.37 0.1480 $(2^{-4}, 0.5)$ (4)	73.68±4.93 0.0689 $(2^{-2}, 0.5)$ (2)	**74.94±4.38** 0.0573 $(2^{2},2^{-2},0.5)$ (1)
ILPD (583 X 10)	r=0	71.36±5.97 0.0216 (2^{-4}) (5)	71.38±5.02 0.0099 $(2^{-4}, 2^{1})$ (4)	71.90±5.46 0.0327 $(2^{3}, 0.2)$ (3)	**73.00±5.55** 0.0378 $(2^{-2}, 0.1)$ (1)	71.95±6.42 0.0018 $(2^{-4},2^{-4},0.5)$ (2)
	r=0.05	71.34±5.65 0.0224 (2^{-2}) (5)	71.54±6.64 0.0087 $(2^{-4}, 2^{1})$ (2)	71.53±6.67 0.0335 $(2^{1}, 0.8)$ (3)	71.38±5.02 0.0380 $(2^{-4}, 0.1)$ (4)	**71.56±6.38** $(2^{-4},2^{-3},0.2)$ 0.0036 (1)
	r=0.1	71.28±6.43 0.0294 (2^{-2})	70.68±7.08 0.0073 $(2^{-2}, 2^{3})$	70.10±5.80 0.0387 $(2^{4}, 0.8)$	**71.54±6.64** 0.0396 $(2^{-3}, 0.2)$	71.36±6.08 0.0051 $(2^{-4},2^{-4},0.5)$

		(3)	(4)	(5)	(1)	(2)
Bupa (345 X 7)	*r*=0	67.26±8.98 0.0084 (2^{4}) (5)	67.41±6.71 0.0064 (2^{-3}, 2^{1}) (4)	67.85±2.48 0.0132 (2^{-4}, 0.5) (3)	68.01±4.58 0.0104 (2^{3}, 0.5) (2)	**68.94±7.30** 0.0018 (2^{-3},2^{2},0.5) (1)
	r=0.05	66.08±4.64 0.0012 (2^{4}) (5)	67.25±4.34 0.0054 (2^{-2}, 2^{3}) (3)	67.00±2.28 0.0159 (2^{1}, 1) (4)	67.89±5.63 0.0143 (2^{4}, 0.2) (2)	**68.31±6.88** 0.0036 (2^{4}, 2^{0},0.8) (1)
	r=0.1	66.72±4.82 0.0096 (2^{4}) (3)	66.35±3.76 0.0023 (2^{-4}, 2^{4}) (5)	66.52±3.51 0.0186 (2^{2}, 0.5) (4)	66.84±5.80 0.0136 (2^{4}, 0.1) (2)	**67.04±5.04** 0.0014 (2^{3}, 2^{-2},0.8) (1)
QSAR (1055 X 41)	*r*=0	**84.83±6.33** 0.0904 (2^{4}) (1)	84.07±5.78 0.0386 (2^{1}, 2^{2}) (5)	84.10±7.96 0.0986 (2^{-3}, 0.1) (4)	84.19±5.66 0.0499 (2^{2}, 0.1) (3)	84.26±5.66 0.0346 (2^{0},2^{0},0.5) (2)
	r=0.05	82.85±8.91 0.0926 (2^{4}) (5)	83.02±7.72 0.0344 (2^{1}, 2^{1}) (4)	83.25±2.90 0.0974 (2^{-1}, 0.2) (3)	83.34±6.29 0.0474 (2^{-2}, 0.1) (2)	**83.54±6.36** 0.0330 (2^{-4}, 2^{4},0.8) (1)
	r=0.1	80.76±9.02 0.0958 (2^{4}) (5)	82.28±9.34 0.0395 (2^{-2}, 2^{-3}) (3)	82.40±5.98 0.0994 (2^{-2}, 0.2) (2)	82.19±6.66 0.0457 (2^{4}, 0.1) (4)	**84.20±3.83** 0.0369 (2^{-3}, 2^{3},0.5) (1)
Rice (3810 X 8)	*r*=0	90.85±3.80 2.2010 (2^{-4}) (5)	91.44±2.12 0.6965 (2^{2}, 2^{4}) (2)	90.94±2.11 2.9128 (2^{-4}, 0.1) (4)	90.99±2.88 0.7368 (2^{-4}, 0.8) (3)	**91.69±2.28** 0.7213 (2^{3}, 2^{1}, 0.2) (1)
	r=0.05	90.63±2.48 2.2070 (2^{-4}) (3)	90.56±2.83 0.6922 (2^{3}, 2^{4}) (5)	90.62±3.43 2.9161 (2^{-4}, 0.5) (4)	90.78±3.25 0.7375 (2^{-4}, 1) (2)	**91.48±2.41** 0.7222 (2^{2}, 2^{4}, 0.8) (1)
	r=0.1	90.05±3.52 2.2081 (2^{-3}) (5)	90.27±2.69 0.6983 (2^{1}, 2^{3}) (4)	90.36±2.31 2.9188 (2^{-3},0.8) (3)	90.53±2.85 0.7398 (2^{-3}, 0.5) (2)	**91.05±3.11** 0.7299 (2^{2}, 2^{1},1) (1)
	r=0	71.45±5.61	73.95±3.09	71.79±4.92	71.97±3.74	**74.20±3.83**

Monks1 (556 X 7)		0.0123 (2^{1}) (5)	0.0062 $(2^{0}, 2^{-4})$ (2)	0.0265 $(2^{-4}, 0.1)$ (4)	0.0142 $(2^{4}, 0.8)$ (3)	0.0010 $(2^{-3},2^{4},0.2)$ (1)
	r=0.05	70.57±4.63 0.0137 (2^{-2}) (4)	73.58±4.77 0.0089 $(2^{0}, 2^{4})$ (2)	70.51±3.46 0.0281 $(2^{0}, 0.8)$ (5)	71.06±2.06 0.0158 $(2^{4}, 1)$ (3)	**73.82±4.71** 0.0045 $(2^{-2},2^{4},0.8)$ (1)
	r=0.1	70.12±5.94 0.0108 (2^{-2}) (5)	72.99±4.86 0.0056 $(2^{-1}, 2^{-4})$ (2)	70.24±4.25 0.0293 $(2^{1}, 0.2)$ (4)	70.34±3.88 0.0120 $(2^{3}, 0.5)$ (3)	**73.12±4.30** 0.0055 $(2^{-3},2^{3},0.5)$ (1)
Monks2 (601 X 7)	*r*=0	65.72±5.68 0.0139 (2^{-4}) (4.5)	66.86±6.23 0.0086 $(2^{-1}, 2^{-4})$ (2)	65.75±6.82 0.0246 $(2^{-4}, 0.1)$ (3)	65.72±6.86 0.0097 $(2^{3}, 0.8)$ (4.5)	**67.87±6.04** 0.0010 $(2^{-4},2^{4},0.5)$ (1)
	r=0.05	64.32±6.82 0.0163 (2^{-4}) (5)	64.38±6.67 0.0014 $(2^{-1}, 2^{-4})$ (4)	65.72±6.07 0.0272 $(2^{-4}, 0.1)$ (2.5)	65.72±6.20 0.0080 $(2^{4}, 0.2)$ (2.5)	**66.71±5.09** 0.0027 $(2^{-4},2^{4},1)$ (1)
	r=0.1	63.52±5.34 0.0197 (2^{-4}) (5)	64.72±6.82 0.0090 $(2^{-1}, 2^{-4})$ (3)	64.05±6.83 0.0284 $(2^{-4}, 0.1)$ (4)	65.47±6.82 0.0048 $(2^{4}, 0.2)$ (2)	**65.73±6.57** 0.0072 $(2^{-4},2^{4},1)$ (1)
Monks3 (554 X 7)	*r*=0	63.51±6.68 0.0132 (2^{-4}) (5)	64.87±4.34 0.0033 $(2^{0}, 2^{-4})$ (2)	64.52±4.16 0.0268 $(2^{-4}, 0.1)$ (4)	**65.98±5.45** 0.0009 $(2^{1}, 0.2)$ (1)	64.55±5.31 0.0047 $(2^{-1},2^{1},0.8)$ (3)
	r=0.05	62.42±4.16 0.0164 (2^{-3}) (5)	**64.77±3.75** 0.0012 $(2^{-2}, 2^{0})$ (1)	63.48±5.02 0.0277 $(2^{-3}, 0.2)$ (4)	64.69±4.98 0.0060 $(2^{0}, 0.8)$ (2)	64.53±3.85 0.0083 $(2^{-4},2^{4},1)$ (3)
	r=0.1	61.39±5.97 0.0182 (2^{-4}) (5)	63.65±4.34 0.0029 $(2^{0}, 2^{3})$ (3)	62.92±4.96 0.0289 $(2^{-2}, 0.5)$ (4)	**64.01±3.86** 0.0015 $(2^{3}, 0.8)$ (1)	63.93±8.65 0.0091 $(2^{-3},2^{4},1)$ (2)

In Table 4.2, the performances of the algorithms by SVM, TBSVM, Pin-SVM, Pin-TSVM and L1-Pin-TBSVM for linear kernel are presented. In comparison with the remaining algorithms, L1-Pin-TBSVM achieves the highest classification accuracy at all levels of noise on 8 datasets among the 11 datasets considered. Also, out of 33 cases (11 datasets x 3 noise levels) the proposed method yields the highest accuracy in 27 cases. These observations indicate the superiority of L1-Pin-TBSVM. Among the remaining algorithms, Pin-TSVM shows in general better performance. Notice that as the quantity of noise rises, the classification accuracy performance of the methods deteriorates. In Table 4.3, the average ranks of the five methods for linear kernel are tabulated for $r = 0$, $r = 0.05$ and $r = 0.1$. Note that smaller average rank means better prediction ability. From the results of Table 4.3, one can see that pinball twin SVMs perform better than hinge loss based SVM and TBSVM. When $r = 0.05$ and $r = 0.1$, enhanced performance of L1-Pin-TBSVM compared to other SVMs clearly illustrates its effectiveness and usefulness on noisy data.

Table 4.3 Average ranks on the accuracy of SVM, TBSVM, Pin-SVM, Pin-TSVM and L1-Pin-TBSVM for linear kernel on benchmark datasets with different levels of noise.

Level of noise	SVM	TBSVM	Pin-SVM	Pin-TSVM	L1-Pin-TBSVM
r=0	4.0909	3.2727	3.6363	2.5909	1.409
r=0.05	4.3636	3.4545	3.5	2.4545	1.2272
r=0.1	4.2727	3.6363	3.4545	2.4545	1.1818

In Table 4.4, the results of SVM, TBSVM, Pin-SVM, Pin-TSVM and L1-Pin-TBSVM using the Gaussian kernel on 11 datasets are shown. One can observe that, at all levels of noise, L1-Pin-TBSVM achieved the highest accuracy performance on 5 datasets. Also, out of 33 cases, it achieved 25 times the highest classification accuracy. This suggests the effectiveness of L1-Pin-TBSVM for Gaussian kernel.

Table 4.4 A comparison of performance of the proposed L1-Pin-TBSVM with SVM, TBSVM, Pin-SVM, Pin-TSVM on benchmark datasets with noise level r. Gaussian kernel was used. Best result is indicated in boldface.

Dataset (Total size)	Level of noise	SVM Time (C, μ) Rank	TBSVM Time $(C1, C2, \mu)$ Rank	Pin-SVM Time (C, μ, τ) Rank	Pin-TSVM Time $(C1, \mu, \tau)$ Rank	L1-Pin-TBSVM Time $(C1, C2, \mu, \tau)$ Rank
Sonar	r=0	90.40±4.99 0.0140 (2^2, 2^0)	89.50±5.06 0.0063 (2^{-4}, 2^{-2}, 2^0)	90.88±7.65 0.0255 (2^1, 2^{-1}, 0.5)	90.88±5.68 0.0095 (2^{-4}, 2^0, 0.5)	**91.85±7.80** 0.0069 (2^4, 2^3, 2^{-1},1)

(208×60)		(4)	(5)	(2.5)	(2.5)	(1)
	r=0.05	88.90±6.45 0.0158 $(2^{3}, 2^{2})$ (3)	87.95±5.73 0.0072 $(2^{-4}, 2^{-1}, 2^{0})$ (5)	88.45±7.19 0.0245 $(2^{2}, 2^{0}, 0.8)$ (4)	89.33±7.32 0.0083 $(2^{-3}, 2^{0}, 0.8)$ (2)	**89.90±5.75** 0.0052 $(2^{4}, 2^{4}, 2^{-4},1)$ (1)
	r=0.1	88.00±6.79 0.0169 $(2^{1}, 2^{-1})$ (2)	87.47±7.00 0.0082 $(2^{-4}, 2^{-2}, 2^{0})$ (3.5)	87.04±6.70 0.0244 $(2^{3}, 2^{-1}, 0.2)$ (5)	87.47±7.10 0.0099 $(2^{-3}, 2^{0}, 0.8)$ (3.5)	**88.95±7.44** 0.0075 $(2^{4}, 2^{4}, 2^{-1},1)$ (1)
Breast-cancer (683×9)	*r*=0	96.79±3.40 0.0993 $(2^{0}, 2^{1})$ (5)	97.37±1.64 0.0461 $(2^{1}, 2^{-4}, 2^{4})$ (2)	96.92±3.66 0.0862 $(2^{2}, 2^{2}, 0.8)$ (4)	96.94±3.57 0.0582 $(2^{-1}, 2^{3}, 0.1)$ (3)	**97.66±2.07** 0.0461 $(2^{2}, 2^{1}, 2^{0},1)$ (1)
	r=0.05	95.89±2.47 0.0953 $(2^{3}, 2^{2})$ (5)	**97.20±2.13** 0.0457 $(2^{1}, 2^{-4}, 2^{4})$ (1)	96.46±2.80 0.0838 $(2^{0}, 2^{2}, 0.5)$ (4)	96.48±3.18 0.0574 $(2^{-2}, 2^{3}, 0.2)$ (3)	97.07±2.07 0.0432 $(2^{3},2^{2},2^{0},0.8)$ (2)
	r=0.1	95.32±2.73 0.0998 $(2^{4}, 2^{0})$ (4)	96.48±1.97 0.0496 $(2^{-4}, 2^{-4}, 2^{1})$ (2)	95.87±2.40 0.0863 $(2^{0}, 2^{3}, 0.5)$ (3)	94.45±3.20 0.0537 $(2^{-2}, 2^{1}, 0.5)$ (5)	**97.37±2.35** 0.0408 $(2^{4},2^{4},2^{-3},0.8)$ (1)
German (1000×24)	*r*=0	75.90±4.22 0.2866 $(2^{3}, 2^{2})$ (5)	76.1±4.72 0.1391 $(2^{-4}, 2^{-3}, 2^{2})$ (4)	76.8±4.15 1.4920 $(2^{4}, 2^{3}, 0.1)$ (2)	76.2±4.73 0.1562 $(2^{0}, 2^{4}, 1)$ (3)	**77.40±4.14** 0.1417 $(2^{2},2^{1},2^{0},1)$ (1)
	r=0.05	74.20±3.85 0.2874 $(2^{2}, 2^{1})$ (5)	75.20±4.44 0.1399 $(2^{-3}, 2^{-2}, 2^{2})$ (3)	75.35±4.24 0.9094 $(2^{4}, 2^{1}, 0.1)$ (2)	74.40±3.30 0.1581 $(2^{1}, 2^{4}, 0.8)$ (4)	**76.30±3.05** 0.1481 $(2^{2},2^{1},2^{0},1)$ (1)
	r=0.1	73.20±2.85 0.2856 $(2^{4}, 2^{2})$	74.40±3.37 0.1352 $(2^{-4}, 2^{-2}, 2^{1})$	74.20±4.10 1.7433 $(2^{3}, 2^{1}, 0.8)$	74.80±4.36 0.1534 $(2^{0}, 2^{4}, 0.5)$	**74.90±4.72** 0.1439 $(2^{2},2^{1},2^{0},1)$

		(5)	(3)	(4)	(2)	(1)
CMC (1473 X 9)	*r*=0	75.02±4.89 1.0301 $(2^{2}, 2^{0})$ (4)	75.09±4.98 0.3678 $(2^{3}, 2^{-3}, 2^{-4})$ (2.5)	75.09±4.89 3.0703 $(2^{3}, 2^{0}, 0.1)$ (2.5)	74.67±2.64 0.4721 $(2^{0}, 2^{3}, 0.2)$ (5)	**75.22±4.92** 0.2583 $(2^{-3},2^{-1},2^{-1},0.8)$ (1)
	r=0.05	74.81±2.80 1.0517 $(2^{2}, 2^{0})$ (3)	74.95±4.93 0.3687 $(2^{4}, 2^{-4}, 2^{-3})$ (2)	74.60±2.79 3.3879 $(2^{0}, 2^{-1}, 0.5)$ (4)	74.47±3.12 0.4792 $(2^{1}, 2^{3}, 0.1)$ (5)	**75.01±2.92** 0.2558 $(2^{-4},2^{-1},2^{-1},1)$ (1)
	r=0.1	74.19±3.15 1.0649 $(2^{0}, 2^{-1})$ (5)	74.75±2.61 0.3635 $(2^{2}, 2^{-4}, 2^{-3})$ (2)	74.26±3.64 3.5635 $(2^{4}, 2^{0}, 0.2)$ (4)	74.40±3.56 0.4756 $(2^{1}, 2^{3}, 0.2)$ (3)	**74.97±4.09** 0.2572 $(2^{2},2^{1},2^{-2},1)$ (1)
ILPD (583 X 10)	*r*=0	72.56±5.97 0.0893 $(2^{-4}, 2^{-1})$ (2.5)	72.36±5.97 0.01120 $(2^{-4}, 2^{1}, 2^{4})$ (5)	72.52±6.25 0.0978 $(2^{-1}, 2^{-3}, 0.8)$ (4)	**72.69±8.02** 0.0182 $(2^{-4}, 2^{4}, 0.8)$ (1)	72.56±5.97 0.0142 $(2^{-4},2^{-2},2^{4},0.8)$ (2.5)
	r=0.05	72.35±5.75 0.0895 $(2^{-2}, 2^{-1})$ (3)	72.01±4.77 0.01185 $(2^{-4}, 2^{1}, 2^{4})$ (5)	**72.38±5.75** 0.0928 $(2^{-2}, 2^{-3}, 0.5)$ (1.5)	72.34±5.57 0.0013 $(2^{-2}, 2^{3}, 0.5)$ (4)	**72.38±5.75** 0.0118 $(2^{-4},2^{-4},2^{-4},0.1)$ (1.5)
	r=0.1	72.35±6.96 0.0865 $(2^{-2}, 2^{0})$ (3)	72.35±4.91 0.01172 $(2^{-2}, 2^{3}, 2^{3})$ (3)	72.35±6.96 0.0960 $(2^{-1}, 2^{-4}, 0.5)$ (3)	71.50±6.73 0.0014 $(2^{-3}, 2^{3}, 0.2)$ (5)	**72.38±6.96** 0.0103 $(2^{-4},2^{-4},2^{-4},0.1)$ (1)
Bupa (345 X 7)	*r*=0	69.62±7.06 0.0394 $(2^{3}, 2^{-1})$ (2)	**69.84±5.80** 0.0090 $(2^{-3}, 2^{1}, 2^{3})$ (1)	69.01±8.91 0.0404 $(2^{4}, 2^{-1}, 0.1)$ (4)	68.43±6.37 0.0172 $(2^{2}, 2^{3}, 0.1)$ (5)	69.52±6.75 0.0026 $(2^{-2},2^{2},2^{-4},0.8)$ (3)
	r=0.05	67.79±6.42 0.0382 $(2^{4}, 2^{0})$	68.69±5.67 0.0094 $(2^{-2}, 2^{3}, 2^{0})$	68.37±6.91 0.0417 $(2^{0}, 2^{-1}, 0.5)$	**68.97±6.61** 0.0158 $(2^{-1}, 2^{3}, 0.1)$	68.43±6.96 0.0066 $(2^{2}, 2^{3}, 2^{-4},1)$

		(5)	(2)	(4)	(1)	(3)
	r=0.1	66.73±8.17 0.0354 $(2^{3}, 2^{-1})$ (5)	67.83±4.61 0.0023 $(2^{-4}, 2^{4}, 2^{-1})$ (3)	67.26±6.83 0.0475 $(2^{4}, 2^{-1}, 0.2)$ (4)	**68.14±8.71** 0.0179 $(2^{-3}, 2^{3}, 0.1)$ (1)	67.92±6.89 0.0094 $(2^{3}, 2^{4}, 2^{-3},1)$ (2)
QSAR (1055 X 41)	*r*=0	85.68±3.35 0.1549 $(2^{4}, 2^{0})$ (3)	84.46±4.79 0.1338 $(2^{0}, 2^{3}, 2^{3})$ (5)	84.63±3.77 1.2182 $(2^{0}, 2^{-2}, 0.8)$ (4)	85.79±3.78 0.1401 $(2^{2}, 2^{3}, 0.1)$ (2)	**86.59±3.60** 0.1280 $(2^{3},2^{4},2^{1},0.5)$ (1)
	r=0.05	84.92±2.67 0.1516 $(2^{2}, 2^{1})$ (3)	84.18±4.56 0.1319 $(2^{0}, 2^{3}, 2^{3})$ (5)	84.35±2.54 1.2075 $(2^{-3}, 2^{-2}, 0.2)$ (4)	85.21±3.40 0.1435 $(2^{-1}, 2^{3}, 0.2)$ (2)	**85.31±3.14** 0.1228 $(2^{3}, 2^{4}, 2^{-3},0.8)$ (1)
	r=0.1	83.79±3.74 0.1572 $(2^{-1}, 2^{0})$ (5)	83.98±4.56 0.1304 $(2^{1}, 2^{4}, 2^{3})$ (4)	84.65±3.94 1.2108 $(2^{-2}, 2^{-1}, 0.2)$ (3)	85.12±2.92 0.1428 $(2^{2}, 2^{3}, 0.1)$ (2)	**85.97±3.38** 0.1246 $(2^{-4}, 2^{-4}, 2^{-4},0.1)$ (1)
Rice (3810 X 8)	*r*=0	91.52±3.09 3.8328 $(2^{-4}, 2^{-4})$ (3)	91.93±3.73 2.4294 $(2^{-2}, 2^{3}, 2^{2})$ (2)	91.41±3.24 3.2793 $(2^{-4}, 2^{-4}, 0.2)$ (4)	91.18±3.29 2.8763 $(2^{1}, 2^{4}, 1)$ (5)	**92.38±3.60** 2.3104 $(2^{2}, 2^{2}, 2^{-4},0.5)$ (1)
	r=0.05	91.46±1.14 3.8322 $(2^{-3}, 2^{-4})$ (2)	91.44±2.12 2.4216 $(2^{-4}, 2^{1}, 2^{4})$ (3)	91.07±1.33 3.2729 $(2^{-1}, 2^{-4}, 0.1)$ (5)	91.18±1.79 2.8731 $(2^{0}, 2^{-4}, 0.8)$ (4)	**92.13±2.75** 2.5493 $(2^{1}, 2^{2}, 2^{-4},0.8)$ (1)
	r=0.1	91.28±1.15 3.8302 $(2^{-2}, 2^{-4})$ (5)	91.59±2.75 2.4210 $(2^{-1}, 2^{2}, 2^{3})$ (4)	91.73±1.32 3.2648 $(2^{-4}, 2^{-3}, 0.5)$ (2)	91.62±1.43 2.8706 $(2^{1}, 2^{3}, 0.5)$ (3)	**91.95±2.07** 2.3849 $(2^{1}, 2^{1}, 2^{-4},0.8)$ (1)
Monks1	*r*=0	85.03±5.37 0.1036 $(2^{3}, 2^{-1})$	**87.70±5.32** 0.0153 $(2^{-4}, 2^{-3}, 2^{-1})$	87.40±5.90 0.0142 $(2^{4}, 2^{-1}, 0.2)$	86.65±2.89 0.0249 $(2^{-4}, 2^{1}, 0.1)$	87.50±8.98 0.0109 $(2^{1},2^{2},2^{-4},1)$

(556 X 7)		(5)	(1)	(3)	(4)	(2)
	r=0.05	84.98±3.59 0.1080 $(2^{4}, 2^{-1})$ (5)	85.90±3.06 0.0157 $(2^{-3}, 2^{-4}, 2^{-2})$ (2)	85.34±2.71 0.0138 $(2^{3}, 2^{-1}, 0.1)$ (3.5)	85.34±5.44 0.0213 $(2^{-2}, 2^{2}, 0.8)$ (3.5)	**86.25±5.08** 0.0152 $(2^{1},2^{2},2^{-3},0.8)$ (1)
	r=0.1	83.15±2.63 0.1056 $(2^{4}, 2^{-1})$ (3)	82.41±4.18 0.0178 $(2^{-3}, 2^{-4}, 2^{0})$ (4)	81.50±4.25 0.0157 $(2^{3}, 2^{0}, 0.1)$ (5)	84.34±5.44 0.0225 $(2^{-4}, 2^{2}, 0.2)$ (2)	**85.87±5.75** 0.0142 $(2^{1},2^{2},2^{-3},0.5)$ (1)
Monks2 (601 X 7)	*r*=0	94.50±6.33 0.0269 $(2^{4}, 2^{-1})$ (3)	93.83±5.24 0.0128 $(2^{-3}, 2^{-4}, 2^{-2})$ (5)	94.37±6.79 0.0128 $(2^{-4}, 2^{0}, 0.5)$ (4)	94.51±3.83 0.0192 $(2^{-4}, 2^{1}, 0.1)$ (2)	**95.83±3.17** 0.0110 $(2^{4},2^{3},2^{1},0.8)$ (1)
	r=0.05	93.19±7.50 0.0277 $(2^{3}, 2^{-1})$ (5)	93.47±5.62 0.0130 $(2^{-1}, 2^{-3}, 2^{-2})$ (4)	93.90±4.55 0.0130 $(2^{2}, 2^{-2}, 0.5)$ (3)	93.95±4.43 0.0185 $(2^{0}, 2^{-4}, 0.8)$ (2)	**94.18±2.66** 0.0125 $(2^{1},2^{2},2^{-2},0.8)$ (1)
	r=0.1	92.87±6.82 0.0206 $(2^{2}\ 2^{-2})$ (5)	93.01±4.73 0.0145 $(2^{1}, 2^{-1}, 2^{-2})$ (4)	**93.66±5.49** 0.0145 $(2^{2}, 2^{-1}, 0.1)$ (1)	93.17±2.77 0.0184 $(2^{1}, 2^{3}, 0.5)$ (3)	93.35±5.90 0.0138 $(2^{2},2^{2},2^{-2},0.5)$ (2)
Monks3 (554 X 7)	*r*=0	63.42±4.58 0.0717 $(2^{-4}\ 2^{-4})$ (4)	63.52±4.42 0.0173 $(2^{-2}, 2^{2}, 2^{2})$ (3)	62.52±4.16 0.0073 $(2^{-4}, 2^{0}, 0.5)$ (5)	63.86±5.76 0.0162 $(2^{3}, 2^{2}, 1)$ (2)	**64.23±3.78** 0.0132 $(2^{4},2^{1},2^{-2},0.5)$ (1)
	r=0.05	61.41±4.53 0.0733 $(2^{-2}\ 2^{-3})$ (5)	62.00±4.48 0.0165 $(2^{-1}, 2^{4}, 2^{2})$ (4)	63.27±3.85 0.0043 $(2^{-2}, 2^{-1}, 0.2)$ (3)	63.86±6.79 0.0121 $(2^{1}, 2^{1}, 0.2)$ (2)	**64.25±3.15** 0.0112 $(2^{4},2^{3},2^{-1},0.8)$ (1)
	r=0.1	61.12±4.16 0.0731 $(2^{-1}\ 2^{-2})$	61.83±5.36 0.0188 $(2^{3}, 2^{-1}, 2^{4})$	63.66±4.52 0.0068 $(2^{-4}, 2^{1}, 0.1)$	**63.85±5.01** 0.0188 $(2^{-4}, 2^{1}, 0.1)$	63.74±4.49 0.0151 $(2^{2},2^{3},2^{-2},1)$

		(5)	(4)	(3)	(1)	(2)

Table 4.5 Average ranks on the accuracy of SVM, TBSVM, Pin-SVM, Pin-TSVM and L1-Pin-TBSVM for Gaussian on benchmark datasets with different levels of noise.

Level of noise	SVM	TBSVM	Pin-SVM	Pin-TSVM	L1-Pin-TBSVM
r=0	3.6818	3.2272	3.5454	3.1363	1.409
r=0.05	4.0	3.2727	3.4545	2.9545	1.3181
R=0.1	4.2727	3.3181	3.3636	2.7727	1.2727

In Table 4.5, the average ranks of the five methods are tabulated for $r = 0$, $r = 0.05$ and $r = 0.1$. One can see that, as the level of noise increases, the average rank for the hinge loss based SVM and TBSVM also increases whereas for pinball loss based SVMs it decreases indicating its advantage in terms of robustness on noisy data. Finally, from the enhanced performance of L1-Pin-TBSVM for $r = 0.05$ and $r = 0.1$ we conclude its superiority for problems whose datasets are corrupted by noise.

The following is a summary of the experimental results in Table 4.2-Table 4.5: (i). Gaussian kernel takes longer to learn than the linear kernel; (ii). slower learning speed result by Pin-SVM could be attributed to solving large sized QPPs; (iii).on average, the learning speed of the proposed L1-Pin-TBSVM is faster than the remaining algorithms; (iv). Gaussian kernel has a higher classification accuracy than the linear kernel.; (v).as the level of noise increases, classification accuracy generally decreases; (vi).Pin-TSVM and L1-Pin-TBSVM show better average ranks than the remaining algorithms; (vii).for noisy data, the lowest average rank by L1-Pin-TBSVM indicates its superiority and robustness.

It is interesting to statistically compare the performance on the classification accuracy of the five algorithms $(k = 5)$ on the 11 datasets $(N = 11)$ for the Gaussian kernel. For this purpose, we proceed in performing statistical tests for r = 0, r = 0.05 and r = 0.1 in that order. In this work, we employ the non-parametric Friedman test which is a simple, robust test with its corresponding Nemenyi post-hoc test (Demsar, 2006). Under the null hypothesis that all the algorithms are equivalent, we consider the Friedman statistic distributed according to χ_F^2 −distribution with $(k - 1)$ degrees of freedom and a better statistic F_F, distributed according to F −distribution with $((k - 1), (k - 1)(N - 1))$ degrees of freedom defined by

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right] \text{ and } F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{4.16}$$

where R_j, k and N denote the average rank of the jth algorithm, the total number of algorithms and datasets respectively. When the null hypothesis is rejected, we proceed for pair-wise comparison of the algorithms with the Nemenyi test by computing the critical difference. For details, see (Demsar, 2006).

From Table 4.5 on the average ranks for the Gaussian kernel where k = 5 and N = 11, we perform the Friedman test and Nemenyi post hoc test. For *r=0,* we get from Table 4.5 and using (24), $\chi_F^2 = 14.7932$ and $F_F = 5.065$. Since the critical value of $F(4{,}40) = 2.606$ for the level of significance $\alpha = 0.05$ is such that $F_F > 2.606$ and hence we reject the null hypothesis. Subsequently, we proceed with pair-wise difference (CD) at $p = 0.10$ is $2.459\sqrt{\frac{k(k+1)}{6N}} \approx 1.6579$ (Demsar, 2006). From Table 4.5, the distinction between: (i). L1-Pin-TBSVM and the best among the rest of the methods is: $3.1363 - 1.409 = 1.7273 > 1.6579(CD),$ we say that L1-Pin-TBSVM performs significantly better than the rest; (ii). the finest and the worst of SVM, TBSVM, Pin-SVM and Pin-TSVM is 3.6818-3.1363= $0.5455 < 1.6579,$ we say that the post-hoc test is not powerful enough to find any significant differences between the algorithms.

Next, we employ the statistical tests for comparison of algorithms on datasets contaminated by noise with *r*=0.05. By the same manner as above, from the average ranks of Table 4.5, we compute $\chi_F^2 = 18.0871$ and $F_F = 6.98$. We see that $F_F > F(4{,}40)$ and hence we reject the null hypothesis. By the Nemenyi post-hoc test, the difference between the average ranks of: (i). L1-Pin-TBSVM and Pin-TSVM are 2.9545 – 1.3181 = 1.6364 < 1.6579, the post-hoc test is insufficiently powerful to distinguish between the two algorithms; (ii). L1-Pin-TBSVM and the best of SVM, TBSVM, Pin-SVM is $3.2727 - 1.3181 = 1.9546 > 1.6579$, we say that L1-Pin-TBSVM performs significantly better than SVM, TBSVM and Pin-SVM.

Finally, the values of χ_F^2 and F_F on noise-corrupted datasets with r = 0.1 can be computed as 21.5041 and 9.5591 respectively. Since $F_F > F(4{,}40)$, we reject the null hypothesis and hence we perform Nemenyi test. In terms of the average ranks of Table 4.5, the difference between: (i). L1-Pin-TBSVM and Pin-TSVM is 2.7727 – 1.2727 = 1.5 < 1.6579, there were no significant differences between the two methods in the post-hoc test; (ii). L1-Pin-TBSVM and the best of SVM, TBSVM, Pin-SVM is $3.3181 - 1.2727 = 2.0454 > 1.6579$ implies L1-Pin-TBSVM performs significantly better than SVM, TBSVM and Pin-SVM.

In light of the statistical comparisons of the algorithms on datasets with three levels of noise presented above, we see that L1-Pin-TBSVM performs significantly better than the other SVMs.

In summary, we conclude that the proposed L1-Pin-TBSVM is an efficient method showing enhanced classification performance and robustness in the presence of noise and outliers.

4.5 Conclusions

With the aim of obtaining an efficient robust learning model, L_1 −norm based twin bounded support vector machine with pinball loss for data classification is presented in this chapter. Besides the benefit of less sensitivity to noise property of pinball loss, with the application of L_1 −norm for within-class scatter minimization, the proposed method also enjoys robustness to outliers. As a novel approach of solving, by a simple reformulation of the primal problem considered, an equivalent pair of dual QPPs in m variables only is derived (L1-Pin-TBSVM), where m is the number of training vectors. In comparison with the twin bounded support vector machine (TBSVM), the duals of our proposed L1-Pin-TBSVM are free of inverse matrices and the non-linear duals can be obtained from their linear formulations directly by applying the kernel trick. Experiments on *Crossplanes* dataset where two or four outliers were introduced show that the proposed L1-Pin-TBSVM outperforms the other SVM methods in terms of accuracy, confirming its robustness to outliers. In addition, empirical findings based on two-moon synthetic dataset and numerous benchmark datasets with varying levels of noise clearly demonstrate improved generalization ability of L1-Pin-TBSVM at comparable training cost which further confirms its effectiveness and suitability where robustness is a problem of major concern. Though L1-Pin-TBSVM is a simple, efficient learning method, it loses sparsity. Increase in the number of parameters in our L1-Pin-TBSVM is also a concern as the selection of their optimal values is a practical problem which needs attention. As a future research, application of optimization methods for large scale datasets like sequential minimal optimization (SMO), dual coordinate descent (DCD) and successive over-relaxation (SOR) is an important practical problem worthy of consideration. Extension to semi-supervised learning is interesting and will be investigated in our future work.

Chapter 5

L1-norm Based Non-parallel Support Vector Machine Classifier with Pinball Loss

5.1 Introduction

In this chapter, we propose a novel L_1 −norm based non-parallel support vector machine with pinball loss for data classification designed at reducing the effect of noise and outliers present in the data. The problem is formulated involving a regularization term, scatter loss and misclassification loss where the scatter of class of vectors is minimized in L_1 −norm and pinball is used for misclassification. As an elegant equivalent problem in simple form, a pair of quadratic programming problems free of inverse of kernel matrices having m variables in the dual space is constructed where m is the number of input data points. Our proposed method allows a unified framework for the linear and non-linear kernels where kernel trick can be applied directly. Empirical results on synthetic and benchmark datasets demonstrate the effectiveness of the proposed method.

Over the last two decades, support vector machine (SVM) proposed by Vapnik (Vapnik, 2000) has evolved into a strong kernel based machine learning tool for classification and regression problems. The popularity of SVM has grown owing to its mathematical elegance and excellent generalization performance on many real-world problems of applications such as face recognition (Kim et al., 2010), fault diagnosis (Ma et al., 2018), text categorization (Basu, et al., 2003), bioinformatics (Guyon, 2002). SVM for classification seeks an optimal hyperplane by separating the two classes of input data points via maximizing the margin between two parallel hyperplanes. The optimal classifier is learned by solving a quadratic programming problem (QPP) consists of a regularization term and hinge loss term where hinge loss is used for measuring misclassification error. SVM implements structural risk minimization (SRM) principle and thereby gain in generalization performance is achieved.

Though SVM leads to a sparse, global solution and further showing excellent generalization performance, the computational complexity in solving the QPP, i.e.$O(m^3)$ where m is the number of input data points, greatly limits its practical application. As efforts to reduce the learning time complexity, two approaches are popular: (i).by constructing fast SVM learning algorithms like decomposition method (Osuna et al., 1997), sequential minimal optimization (SMO) (Platt, 1999), dual coordinate descent (DCD) (Hsieh, 2008); (ii).by relaxing the parallel requirement of supporting hyperplanes, non-parallel SVM classifiers have been proposed as variants of SVM. The generalized eigenvalue proximal support vector machine (GEPSVM) of Mangasarian & Wild (2006) is the first one such example where non-parallel hyperplanes are determined by solving two generalized eigenvalue problems. Similar in spirit of GEPSVM, Jayadeva et al. (2007) proposed twin support vector machine (TWSVM) for binary classification in which the positive and negative non-parallel hyperplanes are constructed so that each of them is close to its own class of input data points and is at least a unit distance away from the other class of data points. This strategy of constructing two non-parallel hyperplanes leads to solving a pair of smaller sized SVM-typed QPPs resulting significant improvement in the training time for TWSVM over SVM (Jayadeva et al., 2007). However, TWSVM implements the empirical risk minimization principle and further it loses sparsity. Due to its reduced computational learning cost and comparable generalization performance, variants of TWSVM and its extensions have been developed in the literature (Balasundaram et al.,2017; Kumar & Gopal, 2009; Peng, 2011; Peng et al., 2016; Shao et al., 2011). Specifically, the least-squares twin SVM (LSTSVM) (Kumar & Gopal, 2009), twin bounded SVM (TBSVM) (Shao et al., 2011) and twin parametric-margin SVM (TPMSVM) (Peng, 2011; Hao, 2010) are some of the successful improvements of TWSVM. Finally for the study of a general framework on non-parallel support vector machines consisting of a regularization term, scatter loss term and misclassification term, see (Mehrkanoon et al., 2014).

Since the hinge loss of SVM realizes the maximum distance between the closest data points of the two classes it was observed that once the input data points, especially around the decision boundary, become noisy wrong learned classifier may result. This shows that SVM is sensitive to feature noise and further it is unstable for re-sampling (Huang et al., 2014). As a significant novel approach, Huang et al. (2014) used quantile distance measure to maximize the distance between the two classes of input data points. By applying the pinball loss that is related to quantile distance in place of hinge loss, a new SVM (Pin-SVM) is proposed in (Huang et al., 2014). A nice property possessed by pinball is that it penalizes also the correctly classified points and thereby less sensitive to feature noise around the decision boundary is reported. Though Pin-SVM shows insensitivity to noise and stability to re-sampling, it loses sparsity. By introducing $\varepsilon -$insensitive zone in the formulation of Pin-SVM, a new $\varepsilon -$insensitive zone Pin-SVM model is proposed to show sparsity (Huang et al., 2014). As a modified $(\varepsilon_1, \varepsilon_2) -$ insensitive zone Pin-SVM, an optimal insensitive zone Pin-SVM is proposed having sparsity and small within-class scatter properties in (Rastogi et al., 2018). Since pinball enjoys robustness to noise, by employing it to the well-known twin parametric-margin support vector machine (TPMSVM) for data classification (Peng, 2011), Xu et al. (2016) proposed pinball loss twin support vector machine (Pin-TSVM) as an extension of pinball loss based non-parallel SVM. In (Huang et al., 2014), an asymmetric least squares SVM (aLS-SVM) is proposed where the margin between the classes is measured by considering expectile value that is related to asymmetric squared pinball loss.

The data arising in real-world applications is subject to the presence of not only feature noise but also outliers. While feature noise is often referred to small perturbations, data points that are significantly far away from the majority of the input data points, such as wrongly labelled data points, are taken as outliers. Since the presence of outliers may seriously affect the performance of the learned model, robustness to outliers is an important study in classification. The main approaches suggested in the literature to decrease the effect of outliers are data cleaning, robust algorithm and robust model approaches (Frenay & Verleysen, 2014). Data cleaning is the method of identifying and eliminating outliers from the set of input data points (Ding & Xu). It is worth noting that often it is hard to detect outliers, especially in higher dimensional datasets. The second approach is on robust optimization algorithms aimed at reducing the effect of noise and outliers. For example, re-weighted strategy was proposed in (Suykens et al., 2002) where different weights were assigned to input data points depending on their residuals. Both the approaches require extra computational efforts and hence expensive. The third approach is the introduction of robust loss functions in the learning models. When L_2 −norm is used, the influence of outliers is exaggerated due to squared operation leading to lower accuracy performance. As a consequence, the popular TWSVM and many of its variants become sensitive to outliers. However, L_1 −norm is generally regarded as an effective way to decrease the effect of outliers present in the data and as a result many works in the literature of robust machine learning are based on L_1 −norm (Peng et al., 2016; Gu et al., 2017; Yan et al., 2018). For example, as a robust-promising model, Guo et al., (2017) proposed L_1 −norm twin projection support vector machine (TPSVM-L1) and applied on image classification. In (Yan et al., 2018) a least squares twin bounded support vector machine based on L_1 −norm (L1-LSTBSVM), whose solution is obtained by an iterative method, has been proposed. By simultaneously minimizing L_1 −norm based scatter loss and the misclassification error by hinge loss, L_1 −norm loss based twin support vector machine (L1LTSVM) has been proposed in (Peng et al., 2016). Meanwhile, some researchers argue that the popular hinge loss and pinball loss functions are unbounded and thus they will be sensitive to outliers to some extent. In view of it, truncated non-convex bounded loss functions are proposed as robust learning models (Shen et al., 2017; Yang & Dong, 2018). However, non-convexity introduces difficulty in optimization.

From the above discussion, we summarize that pinball can bring noise insensitivity and stability to re-sampling whereas L_1 −norm distance measure can improve the robustness of the model. To the best of our knowledge, support vector machines that are simultaneously having the above advantages have not been yet formulated. Accordingly, inspired from the studies of

Pin-SVM and Pin-TSVM (Huang et al., 2014; Xu et al., 2016), we present an outlier-robust, non-parallel twin support vector machine learning method (L1-Pin-NPSVM) with noise insensitivity that also preserves the elegant formulation of the classical SVM. The proposed method is evaluated with TPMSVM, Pin-SVM and Pin-TSVM on *Crossplanes* dataset having two and four outliers, the popular Ripley dataset and fifteen benchmark datasets with different levels of noise. Experimental results on the datasets confirm the robustness and effectiveness of the proposed L1-Pin-NPSVM.

The main contributions of this work can be summarized as follows:

- A novel L_1 −norm based non-parallel support vector machine classifier with pinball loss is proposed where for scatter loss and misclassification error, the L_1 −norm and pinball loss are used.
- A pair of QPPs in the dual space with box constraints (L1-Pin-NPSVM) and free of matrix inversion terms with *m* number of variables is solved wher*e m* is the number of input data points.
- The non-linear dual QPPs can be derived directly from its linear problem by applying kernel trick.
- When both the positive and negative classes have equal number of input data points, the size of the dual QPPs corresponding to L1-Pin-NPSVM and Pin-TSVM is the same.
- Empirical results on synthetic datasets with outliers and feature noise and on benchmark datasets contaminated with three different levels of noise confirm the robustness and superiority of the proposed method.

All vectors are assumed as column vectors. For any vector $\boldsymbol{x} = (x_1, \dots, x_n)^t \in R^n$, we represent its transpose by $\boldsymbol{x}^t$. Its L_1 −norm and L_2 −norm is denoted by $||\boldsymbol{x}||_1$ and $||\mathbf{x}||$ respectively. The vectors of zeros and ones of appropriate dimensions are indicated by **0** and **e** correspondingly and further, we denote the identity matrix of appropriate size by *I*.

The rest of the chapter is organized as follows. In Section 5.2, L_1 −norm based non-parallel SVM with pinball loss is introduced and as an equivalent problem in simple form a pair of SVM-typed QPPs in the dual space is derived (L1-Pin-NPSVM) whose noise insensitivity and scatter minimization properties are verified. A comparative study of the proposed method with TPMSVM, Pin-SVM and Pin-TSVM is presented in Section 5.3. Numerical experiments are detailed and their results are reported in Section 5.4 while Section 5.5 concludes the chapter.

5.2 L1-norm Based Non-parallel Support Vector Machine with Pinball Loss

Consider the binary classification problem where the positive and negative classes consist of m_1 and m_2 number of input data points respectively. It has been shown that both Pin-SVM and Pin-TSVM are efficient classification methods having noise insensitivity and stability to re-sampling. In addition to the advantages, as an important contribution on developing a robust learning model, we present a novel outlier-robust L_1 −norm based non-parallel SVM with pinball loss (L1-Pin-NPSVM) in this section. Besides robustness, we show that the proposed method possesses small within-class scatter and misclassification error properties.

5.2.1 Linear L1-Pin-NPSVM

Consider the linear problem seeking two non-parallel hyperplanes of the form

$$f_1(\boldsymbol{x}) = \boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0 \text{ and } f_2(\boldsymbol{x}) = \boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0 \tag{5.1}$$

where $\boldsymbol{w}_1, \boldsymbol{w}_2 \in R^n$ and $b_1, b_2 \in R$ are unknowns. Let them separate the classes of data points such that $f_1(\boldsymbol{x}_i) \geq 0,\ i = 1,2,\ldots,m_1$ and $f_2(\boldsymbol{x}_j) \leq 0\ j = m_1 + 1,\ldots,m_1 + m_2$.

Following the problem formulation of Pin-TSVM (Xu et al., 2016), by computing the misclassification error using pinball loss defined of the form:

$$L_\tau(x, y, f(\boldsymbol{x})) = \begin{cases} 0 - yf(\boldsymbol{x}), & 0 - yf(\boldsymbol{x}) \geq 0, \\ -\tau(0 - yf(\boldsymbol{x})), & 0 - yf(\boldsymbol{x}) < 0 \end{cases} \tag{5.2}$$

and introducing L_1 −norm for scatter loss, it is proposed to obtain the non-parallel hyperplanes (5.1) by solving the following pair of QPPs:

$$\min_{w_1,b_1} \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2||_1 + c_1\sum_{i=1}^{m_1} L_{\tau_1}(\boldsymbol{x}_i, y_i, f_1(\boldsymbol{x}_i)) \tag{5.3a}$$

And

$$\min_{w_2,b_2} \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1||_1 + c_2\sum_{j=m_1+1}^{m_1+m_2} L_{\tau_2}(\boldsymbol{x}_j, y_j, f_2(\boldsymbol{x}_j)). \tag{5.3b}$$

Or equivalently

$$\begin{aligned} &\min_{w_1,b_1,\xi_1,\xi_2} \quad \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||\boldsymbol{\xi}_2||_1 + c_1\boldsymbol{e}_1^t\boldsymbol{\xi}_1 \\ &\text{subject to} \qquad B\mathbf{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2 = \boldsymbol{\xi}_2, \\ &A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 \geq \mathbf{0}_1 - \boldsymbol{\xi}_1, A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 \leq \mathbf{0}_1 + \frac{1}{\tau_1}\boldsymbol{\xi}_1, \end{aligned} \tag{5.4a}$$

and

$$\begin{aligned} \min_{w_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad & \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||\boldsymbol{\eta}_1||_1 + c_2 \boldsymbol{e}_2^t \boldsymbol{\eta}_2 \\ \text{subject to} \quad & A\mathbf{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1 = \boldsymbol{\eta}_1, \\ & -(B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \geq \mathbf{0}_2 - \boldsymbol{\eta}_2, -(B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \leq \mathbf{0}_2 + \frac{1}{\tau_2}\boldsymbol{\eta}_2, \end{aligned} \tag{5.4b}$$

where $\mathbf{0}_k \in R^{m_k}$ is the column vector of zeros, c_k, c_{k+1} are regularization parameters, $\boldsymbol{\xi}_k, \boldsymbol{\eta}_k \in R^{m_k}$ are vectors of slack variables, $\tau_k > 0$ are parameters and $k = 1, 2$.

One can notice that the third term of the objective functions of Pin-TSVM and L1-Pin-NPSVM, namely the misclassification error term using pinball, is exactly the same. However, unlike Pin-TSVM, the bias term is also penalized in L1-Pin-NPSVM. In view of the fact that L_1 −norm helps in reducing the effect of outliers, as a simple and an efficient way to promote robustness (Peng et al., 2016; Yan et al., 2018), the negative class data points are clustered around the parallel positive hyperplane $f_1(\boldsymbol{x}) = -1$ by minimizing $(B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2)$ using L_1 −norm in (5.4a). This leads to the negative class data points to be as far away as possible from the positive hyperplane. Likewise the positive class points are made as close as possible to the parallel negative hyperplane $f_2(\boldsymbol{x}) = 1$ in (5.4b) by minimizing $||A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1||_1$. Clearly both of them correspond to scatter loss minimization in L_1 −norm.

Remark 5.1. When τ_1=τ_2= 0 the second inequality constraints in (5.4a) and (5.4b) will degenerate into $\boldsymbol{\xi}_1 \geq \mathbf{0}$ and $\boldsymbol{\eta}_2 \geq \mathbf{0}$ respectively (Huang et al., 2014). In this case, problem (5.4) becomes

$$\min_{w_1, b_1, \xi_1, \xi_2} \quad \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||\boldsymbol{\xi}_2||_1 + c_1 \boldsymbol{e}_1^t \boldsymbol{\xi}_1$$

$$\text{subject to} \qquad B\mathbf{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2 = \boldsymbol{\xi}_2,$$

$$A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 \geq -\boldsymbol{\xi}_1, \boldsymbol{\xi}_1 \geq \mathbf{0}_1,$$

and

$$\min_{w_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||\boldsymbol{\eta}_1||_1 + c_2 \boldsymbol{e}_2^t \boldsymbol{\eta}_2$$

$$\text{subject to} \quad A\mathbf{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1 = \boldsymbol{\eta}_1,$$

$$-(B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \geq -\boldsymbol{\eta}_2, \boldsymbol{\eta}_2 \geq \mathbf{0}_2.$$

R**emark 5.2.** It is very interesting to observe that when $\tau_1 = \tau_2 = 1$, problem (5.4) becomes

$$\min_{w_1, b_1, \xi_1, \xi_2} \quad \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||\boldsymbol{\xi}_2||_1 + c_1||\boldsymbol{\xi}_1||_1$$

$$\text{subject to} \quad (B\mathbf{w}_1 + b_1\boldsymbol{e}_2) + \boldsymbol{e}_2 = \boldsymbol{\xi}_2,$$

$$A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 = \boldsymbol{\xi}_1,$$

and

$$\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + ||\boldsymbol{\eta}_1||_1 + c_2||\boldsymbol{\eta}_2||_1$$

$$\text{subject to} \quad A\mathbf{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1 = \boldsymbol{\eta}_1,$$

$$B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2 = \boldsymbol{\eta}_2,$$

which is a special case of the general framework of non-parallel SVM formulation of (Mehrkanoon, 2014) where L_1 −norm is used for both the loss and therefore it can be referred as $L_1 - L_1$ loss SVM problem.

On variants of non-parallel support vector classifiers fitting in the general framework of (Mehrkanoon, 2014) where L_1 −norm distance for scatter loss and L_2 −norm for misclassification error are applied, see (Yan et al., 2018); and however for L_1 −norm for scatter loss and hinge loss for misclassification, we refer to (Peng et al., 2016).

As a novel approach of problem solving, we proceed in constructing an equivalent problem formulation of (5.4) in an elegant and simpler form whose duals will have *m* number of unknowns and *m* is the number of input data points. With this objective, problem (5.4) is equivalently written in the following manner:

$$\begin{aligned} &\min_{\boldsymbol{w}_1, b_1, \xi_1, \xi_2} \quad \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + \boldsymbol{e}_2^t\boldsymbol{\xi}_2 + c_1\boldsymbol{e}_1^t\boldsymbol{\xi}_1 \\ &\text{subject to} \quad B\mathbf{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2 \le \boldsymbol{\xi}_2, \;\; -(B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2) \le \boldsymbol{\xi}_2, \\ &\qquad A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1 \le \frac{1}{\tau_1}\boldsymbol{\xi}_1, \;\; -(A\boldsymbol{w}_1 + b_1\boldsymbol{e}_1) \le \boldsymbol{\xi}_1, \end{aligned} \tag{5.5a}$$

and

$$\begin{aligned} &\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2}c_4(||\boldsymbol{w}_2||^2 + b_2^2) + \boldsymbol{e}_1^t\boldsymbol{\eta}_1 + c_2\boldsymbol{e}_2^t\boldsymbol{\eta}_2 \\ &\text{subject to} \quad A\mathbf{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1 \le \boldsymbol{\eta}_1, \;\; -(A\boldsymbol{w}_2 + b_2\boldsymbol{e}_1 - \boldsymbol{e}_1) \le \boldsymbol{\eta}_1, \\ &\qquad B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2 \le \boldsymbol{\eta}_2, \;\; -(B\boldsymbol{w}_2 + b_2\boldsymbol{e}_2) \le \frac{1}{\tau_2}\boldsymbol{\eta}_2 \end{aligned} \tag{5.5b}$$

Or equivalently we have

$$\min_{\boldsymbol{w}_1, b_1, \xi_1, \xi_2} \quad \frac{1}{2}c_3(||\boldsymbol{w}_1||^2 + b_1^2) + \boldsymbol{e}_2^t\boldsymbol{\xi}_2 + c_1\boldsymbol{e}_1^t\boldsymbol{\xi}_1$$

$$\text{subject to} \quad \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} \le \begin{bmatrix} (1/\tau_1)\boldsymbol{\xi}_1 \\ \boldsymbol{\xi}_2 \end{bmatrix} - \begin{bmatrix} \mathbf{0}_1 \\ \boldsymbol{e}_2 \end{bmatrix}, - \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} \le \begin{bmatrix} \boldsymbol{\xi}_1 \\ \boldsymbol{\xi}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0}_1 \\ \boldsymbol{e}_2 \end{bmatrix},$$

and

$$\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2} c_4(||\boldsymbol{w}_2||^2 + b_2^2) + \boldsymbol{e}_1^t \boldsymbol{\eta}_1 + c_2 \boldsymbol{e}_2^t \boldsymbol{\eta}_2$$

$$subject\ to \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_2 \\ b_2 \end{bmatrix} \le \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{0}_2 \end{bmatrix}, \; -\begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_2 \\ b_2 \end{bmatrix} \le \begin{bmatrix} \boldsymbol{\eta}_1 \\ (1/\tau_2)\boldsymbol{\eta}_2 \end{bmatrix} - \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{0}_2 \end{bmatrix},$$

which in matrix form becomes

$$\begin{gathered} \min_{\boldsymbol{z}_1, \boldsymbol{\xi}} \; \frac{1}{2} c_3 ||\boldsymbol{z}_1||^2 + \boldsymbol{e}^t D_{11} \boldsymbol{\xi} \\ \text{subject to} \quad G\boldsymbol{z}_1 \le D_{12}\boldsymbol{\xi} - \boldsymbol{r}_1, \quad -G\boldsymbol{z}_1 \le \boldsymbol{\xi} + \boldsymbol{r}_1 \end{gathered} \tag{5.6a}$$

and

$$\begin{gathered} \min_{\boldsymbol{z}_2, \boldsymbol{\eta}} \; \frac{1}{2} c_4 ||\boldsymbol{z}_2||^2 + \boldsymbol{e}^t D_{22} \boldsymbol{\eta} \\ \text{subject to} \quad G\boldsymbol{z}_2 \le \boldsymbol{\eta} + \boldsymbol{r}_2, \quad -G\boldsymbol{z}_2 \le D_{21}\boldsymbol{\eta} - \boldsymbol{r}_2 \end{gathered} \tag{5.6b}$$

where

$\boldsymbol{z}_k = \begin{bmatrix} \boldsymbol{w}_k \\ b_k \end{bmatrix} \in R^{n+1}$, $G = \begin{bmatrix} A & \boldsymbol{e}_1 \\ B & \boldsymbol{e}_2 \end{bmatrix} = [C \quad \boldsymbol{e}]$, $\boldsymbol{r}_1 = \begin{bmatrix} \boldsymbol{0}_1 \\ \boldsymbol{e}_2 \end{bmatrix}$, $\boldsymbol{r}_2 = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{0}_2 \end{bmatrix}$, $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}_1 \\ \boldsymbol{\xi}_2 \end{bmatrix}$, $\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix}$, $\boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{bmatrix}$; $\boldsymbol{0}_k, \boldsymbol{e}_k \in R^{m_k}$ indicate the column vectors of zeros and ones, and $D_{11} = diag(c_1\boldsymbol{e}_1; \boldsymbol{e}_2)$, $D_{12} = diag((1/\tau_1)\boldsymbol{e}_1; \boldsymbol{e}_2)$,$D_{22} = diag(\boldsymbol{e}_1; c_2\boldsymbol{e}_2)$, $D_{21} = diag(\boldsymbol{e}_1; (1/\tau_2)\boldsymbol{e}_2)$ are diagonal matrices of order *m*.

The solutions of the primal QPPs (5.6) are obtained by solving their duals. By introducing the vectors of Lagrangian multipliers $\boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2 \ge \boldsymbol{0}$ from R^m, the following Lagrangian functions are formulated

$$\tilde{L}_1(\boldsymbol{z}_1, \boldsymbol{\xi}, \boldsymbol{\alpha}_1, \boldsymbol{\beta}_1) = \frac{1}{2} c_3 \boldsymbol{z}_1^t \boldsymbol{z}_1 + \boldsymbol{e}^t D_{11} \boldsymbol{\xi} + \boldsymbol{\alpha}_1^t (G\boldsymbol{z}_1 - D_{12}\boldsymbol{\xi} + \boldsymbol{r}_1) + \boldsymbol{\beta}_1^t (-G\boldsymbol{z}_1 - \boldsymbol{\xi} - \boldsymbol{r}_1)$$

and

$$\tilde{L}_2(\boldsymbol{z}_2, \boldsymbol{\eta}, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2) = \frac{1}{2} c_4 \boldsymbol{z}_2^t \boldsymbol{z}_2 + \boldsymbol{e}^t D_{22} \boldsymbol{\eta} + \boldsymbol{\alpha}_2^t (G\boldsymbol{z}_2 - \boldsymbol{\eta} - \boldsymbol{r}_2) + \boldsymbol{\beta}_2^t (-G\boldsymbol{z}_2 - D_{21}\boldsymbol{\eta} + \boldsymbol{r}_2)$$

Applying the necessary and sufficient KKT optimality conditions, we obtain

$$\partial \tilde{L}_k / \partial \boldsymbol{z}_k = \boldsymbol{0} \Rightarrow \boldsymbol{z}_k = \frac{(-G^t)}{c_{k+2}} (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k) \quad \text{for } k \;=\; 1, 2, \tag{5.7}$$

$$\partial \tilde{L}_1 / \partial \boldsymbol{\xi} = \boldsymbol{0} \Rightarrow D_{12}\boldsymbol{\alpha}_1 + \boldsymbol{\beta}_1 = D_{11}\boldsymbol{e} \text{ and } \partial \tilde{L}_2 / \partial \boldsymbol{\eta} = \boldsymbol{0} \Rightarrow \boldsymbol{\alpha}_2 + D_{21}\boldsymbol{\beta}_2 = D_{22}\boldsymbol{e} \tag{5.8}$$

$$\boldsymbol{\alpha}_1^t (G\boldsymbol{z}_1 - D_{12}\boldsymbol{\xi} + \boldsymbol{r}_1) = 0, \boldsymbol{\beta}_1^t (-G\boldsymbol{z}_1 - \boldsymbol{\xi} - \boldsymbol{r}_1) = 0 \tag{5.9}$$

$$\boldsymbol{\alpha}_2^t (G\boldsymbol{z}_2 - \boldsymbol{\eta} - \boldsymbol{r}_2) = 0, \; \boldsymbol{\beta}_2^t (-G\boldsymbol{z}_2 - D_{21}\boldsymbol{\eta} + \boldsymbol{r}_2) = 0 \text{ and } \boldsymbol{\alpha}_1, \boldsymbol{\beta}_1, \boldsymbol{\alpha}_2, \boldsymbol{\beta}_2 \ge \boldsymbol{0}\,. \tag{5.10}$$

Substituting (5.7) and (5.8) in the Lagrangian functions will lead to the duals of (5.6) as a pair of QPP minimization problems

$$\min_{\boldsymbol{\alpha}_1,\boldsymbol{\beta}_1\in R^m} \frac{1}{2c_3}(\boldsymbol{\alpha}_1-\boldsymbol{\beta}_1)^t GG^t(\boldsymbol{\alpha}_1-\boldsymbol{\beta}_1)-\boldsymbol{r}_1^t(\boldsymbol{\alpha}_1-\boldsymbol{\beta}_1)$$
$$\text{subject to } \ D_{12}\boldsymbol{\alpha}_1+\boldsymbol{\beta}_1=D_{11}\boldsymbol{e} \text{ and } \boldsymbol{\alpha}_1\geq \boldsymbol{0},\boldsymbol{\beta}_1\geq\boldsymbol{0} \tag{5.11a}$$

and

$$\min_{\boldsymbol{\alpha}_2,\boldsymbol{\beta}_2\in R^m} \frac{1}{2c_4}(\boldsymbol{\alpha}_2-\boldsymbol{\beta}_2)^t GG^t(\boldsymbol{\alpha}_2-\boldsymbol{\beta}_2)+\boldsymbol{r}_2^t(\boldsymbol{\alpha}_2-\boldsymbol{\beta}_2)$$
$$\text{subject to } \ \boldsymbol{\alpha}_2+D_{21}\boldsymbol{\beta}_2=D_{22}\boldsymbol{e} \text{ and } \boldsymbol{\alpha}_2\geq \boldsymbol{0},\boldsymbol{\beta}_2\geq\boldsymbol{0}. \tag{5.11b}$$

Since $D_{12}\boldsymbol{\alpha}_1+\boldsymbol{\beta}_1=D_{11}\boldsymbol{e}$, we have $\boldsymbol{\beta}_1=D_{11}\boldsymbol{e}-D_{12}\boldsymbol{\alpha}_1\geq\boldsymbol{0}$ and $\boldsymbol{\alpha}_1-\boldsymbol{\beta}_1=(I+D_{12})\boldsymbol{\alpha}_1-D_{11}\boldsymbol{e}$. Putting this results in the dual problem (5.11a) it can be equivalently expressed as

$$\min_{\boldsymbol{\alpha}_1\in R^m} L_1(\boldsymbol{\alpha}_1)=\frac{1}{2}\boldsymbol{\alpha}_1^tQ_1\boldsymbol{\alpha}_1-\boldsymbol{h}_1^t\boldsymbol{\alpha}_1$$
$$\text{subject to } \ \boldsymbol{0}\leq\boldsymbol{\alpha}_1\leq D_{12}^{-1}D_{11}\boldsymbol{e}=\begin{pmatrix}c_1\tau_1\boldsymbol{e}_1\\ \boldsymbol{e}_2\end{pmatrix}, \tag{5.12a}$$

where $Q_1=(I+D_{12})GG^t(I+D_{12})\ and\ \boldsymbol{h}_1=(I+D_{12})(GG^tD_{11}\boldsymbol{e}+c_3\boldsymbol{r}_1)$.

Similarly, the conditions $\boldsymbol{\alpha}_2+D_{21}\boldsymbol{\beta}_2=D_{22}\boldsymbol{e}$ and $\boldsymbol{\alpha}_2\geq\boldsymbol{0}$ $\Longrightarrow \boldsymbol{\alpha}_2-\boldsymbol{\beta}_2=D_{22}\boldsymbol{e}-(I+D_{21})\boldsymbol{\beta}_2$ and $D_{22}\boldsymbol{e}\geq D_{21}\boldsymbol{\beta}_2$ hold, an equivalent problem of (5.11b) in simple form is obtained

$$\min_{\boldsymbol{\beta}_2\in R^m} L_2(\boldsymbol{\beta}_2)=\frac{1}{2}\boldsymbol{\beta}_2^tQ_2\boldsymbol{\beta}_2-\boldsymbol{h}_2^t\boldsymbol{\beta}_2,$$
$$\text{subject to } \ \boldsymbol{0}\leq\boldsymbol{\beta}_2\leq D_{21}^{-1}D_{22}\boldsymbol{e}=\begin{pmatrix}\boldsymbol{e}_1\\ c_2\tau_2\boldsymbol{e}_2\end{pmatrix} \tag{5.12b}$$

where $Q_2=(I+D_{21})GG^t(I+D_{21})\ \ and\ \ \boldsymbol{h}_2=(I+D_{21})(GG^tD_{22}\boldsymbol{e}+c_4\boldsymbol{r}_2)$.

We call the pair of quadratic minimization problems with linear inequality constraints (5.12a) – (5.12b), the L_1 −norm based pinball loss non-parallel support vector machine (L1-Pin-NPSVM).

From the solutions of (5.12) and using (5.7), the linear classifier for L1-Pin-NPSVM is given by

$$f(\boldsymbol{x})=sign\left(\frac{\boldsymbol{w}_1^t\boldsymbol{x}+b_1}{||\boldsymbol{w}_1||}+\frac{\boldsymbol{w}_2^t\boldsymbol{x}+b_2}{||\boldsymbol{w}_2||}\right).$$

Remark 5.3. It is assumed that $(\tau_1,\tau_2)\neq(0,0)$, in the above derivation of the problem formulation (5.12). Notice that our approach of deriving (5.12) is novel and is applicable for all $\tau_1,\tau_2>0$.

5.2.2 Non-Linear L1-Pin-NPSVM

Our results (5.12) for the linear kernel can be easily extended to the problem of non-linear classifiers. Assuming that the input data points are taken into a higher dimensional feature space via a non-linear mapping $\varphi(.)$, non-linear L1-Pin-NPSVM in the feature space leads to solving

$$\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}_1, \boldsymbol{\xi}_2} \quad \frac{1}{2} c_3 (||\boldsymbol{w}_1||^2 + b_1^2) + ||\boldsymbol{\xi}_2||_1 + c_1 \boldsymbol{e}_1^t \boldsymbol{\xi}_1 \tag{5.13a}$$

and

$$\begin{gathered} \min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2} \quad \frac{1}{2} c_4 (||\boldsymbol{w}_2||^2 + b_2^2) + ||\boldsymbol{\eta}_1||_1 + c_2 \boldsymbol{e}_2^t \boldsymbol{\eta}_2 \\ \text{subject to} \quad \varphi(A)\mathbf{w}_2 + b_2 \boldsymbol{e}_1 - \boldsymbol{e}_1 = \boldsymbol{\eta}_1, \\ -(\varphi(B)\boldsymbol{w}_2 + b_2 \boldsymbol{e}_2) \geq \mathbf{0}_2 - \boldsymbol{\eta}_2, -(\varphi(B)\boldsymbol{w}_2 + b_2 \boldsymbol{e}_2) \leq \mathbf{0}_2 + \frac{1}{\tau_2} \boldsymbol{\eta}_2, \end{gathered} \tag{5.13b}$$

where $\mathbf{0}_k \in R^{m_k}$ is the column vector of zeros, $c_k > 0$ and $c_{k+1} > 0$ are regularization parameters, $\boldsymbol{\xi}_k, \boldsymbol{\eta}_k \in R^{m_k}$ are vectors of slack variables, $\tau_k > 0$ are parameters and $k = 1, 2$.

Clearly, we have $G = \begin{bmatrix} \varphi(A) & \boldsymbol{e}_1 \\ \varphi(B) & \boldsymbol{e}_2 \end{bmatrix} = [\varphi(C) \quad \boldsymbol{e}]$ and hence $GG^t = K(C, C^t) + E$ in which $E = \boldsymbol{e}\boldsymbol{e}^t$ is the matrix where all its entries become 1. By considering the equivalent problem (5.5) for the non-linear case and proceeding as in the linear case, the duals of (5.13) can be derived as a pair of QPPs in kernel based formulation, again of the form (5.12) In this case,

$$\boldsymbol{z}_k = [\varphi(C)\boldsymbol{e}]^t \boldsymbol{u}_k \text{ and } f_k(\boldsymbol{x}) = [\varphi(\boldsymbol{x})1] \begin{bmatrix} \boldsymbol{w}_k \\ b_k \end{bmatrix} = -(K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_k \text{ for } k = 1,2,$$

where $\boldsymbol{u}_1 = ((I + D_{12})\boldsymbol{\alpha}_1 - D_{11}\boldsymbol{e})/c_3$ and $\boldsymbol{u}_2 = (D_{22}\boldsymbol{e} - (I + D_{21})\boldsymbol{\beta}_2)/c_4$.

Finally, from the solutions of the non-linear L1-Pin-NPSVM problem (5.12) and using (5.7) we construct the non-linear classifier as

$$f(\boldsymbol{x}) = sign\left(\frac{(K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_1}{||\boldsymbol{u}_1^t K(C, C^t)\boldsymbol{u}_1||} + \frac{(K(\boldsymbol{x}^t, C^t) + \boldsymbol{e}^t)\boldsymbol{u}_2}{||\boldsymbol{u}_2^t K(C, C^t)\boldsymbol{u}_2||}\right)$$

Clearly, when $K(C, C^t) = CC^t$, problem (5.12) for the non-linear case reduces to the linear problem. This shows that a uniform formulation for the linear and non-linear kernels is guaranteed by L1-Pin-NPSVM and there is no need for deriving different formulations separately for linear and non-linear kernels.

Remark 5.4. The problem (5.12) is a QPP with box constraints in *m* variables and thus the computational complexity of training L1-Pin-NPSVM is $2O(m^3)$.

Remark 5.5. L1-Pin-NPSVM has the advantage that its learning time will be much less than TWSVM and TBSVM since the dual QPPs (5.12a) and (5.12b) do not have inverse of kernel matrices.

Remark 5.6. Using the KKT conditions (5.8) –(5.10), it is simple to verify that $\boldsymbol{\alpha}_1, \boldsymbol{\beta}_2$ become sparse. Thus, from (5.7), we say that the solution vectors $\boldsymbol{w}_1$and $\boldsymbol{w}_2$ are partly sparse (Peng et al., 2016).

Remark 5.7. The dual problems (5.12a) and (5.12b) are linearly constrained QPPs. One can solve them using SVM-type algorithms like sequential minimal optimization (SMO) (Platt, 1999), dual coordinate descent (DCD) (Hsieh et al., 2008). In this work we used MOSEK optimization toolbox of MATLAB.

5.2.3 Properties of L1-Pin-NPSVM

As the main benefits of introducing pinball loss, we analyze the noise insensitivity, within-class scatter and misclassification error properties of L1-Pin-NPSVM. For easy comprehension, we focus on its linear problem in unconstrained form (5.3) As similar arguments hold for the formulation (5.3b), we restrict our discussion on the problem formulation (5.3a).

Let a generalized sign function be defined as

$$sgn_\tau(u) = \begin{cases} 1, & u > 0 \\ [-\tau, 1], & u = 0. \\ -\tau & u < 0 \end{cases}$$

From the optimality condition of (5.3a) we get

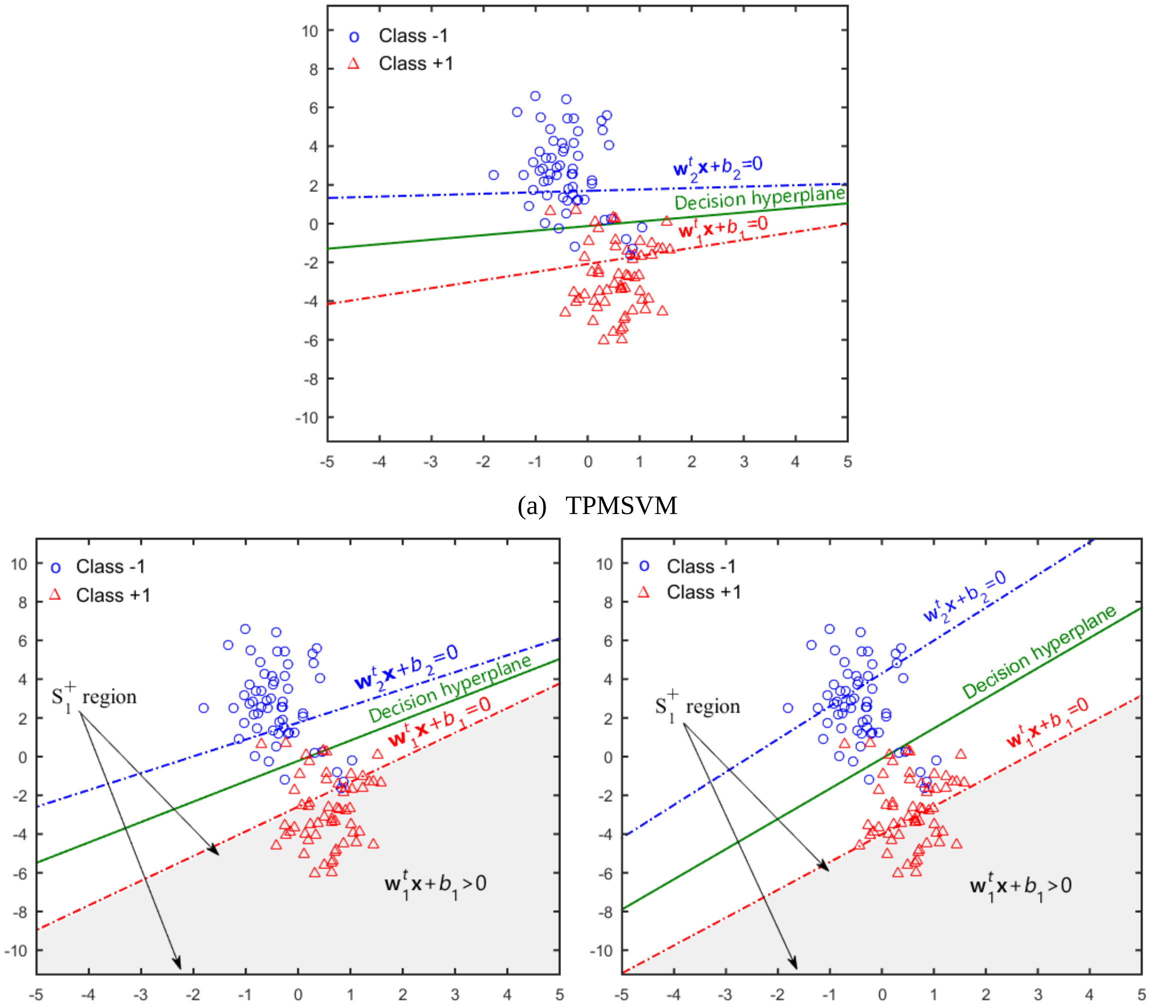
$$\begin{aligned} \mathbf{0} \in \frac{c_3}{c_1}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} + \frac{[B\ \boldsymbol{e}_2]^t}{c_1} sgn_{\tau=1}(B\boldsymbol{w}_1 + b_1\ \boldsymbol{e}_2 + \boldsymbol{e}_2) \\ + \sum_{i=1}^{m_1} sgn_{\tau_1}(-(\boldsymbol{w}_1^t\boldsymbol{x}_i + b_1))\begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix}. \end{aligned} \tag{5.14}$$

Let $S_1^+ = \{i: \boldsymbol{w}_1^t\boldsymbol{x}_i + b_1\ > 0\}$, $S_1^- = \{i: \boldsymbol{w}_1^t\boldsymbol{x}_i + b_1\ < 0\}$ and $S_1^0 = \{i: \boldsymbol{w}_1^t\boldsymbol{x}_i + b_1\ = 0\}$ be a partition of the index set for positive class data points, i.e., $i = 1, \ldots, m_1$. Then, there exists $\varsigma_i \in [-\tau_1, 1]$ such that the condition (5.14) becomes

$$\begin{aligned} \frac{c_3}{c_1}\begin{bmatrix} \boldsymbol{w}_1 \\ b_1 \end{bmatrix} + \frac{[B\ \boldsymbol{e}_2]^t}{c_1} sgn_{\tau=1}(B\boldsymbol{w}_1 + b_1\ \boldsymbol{e}_2 + \boldsymbol{e}_2) \\ -\tau_1 \sum_{i \in S_1^+} \begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix} + \sum_{i \in S_1^-} \begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix} + \sum_{i \in S_1^0} \varsigma_i \begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix} = \mathbf{0} \end{aligned} \tag{5.15}$$

Once $c_1, c_3, \boldsymbol{w}_1$ and b_1 are given, from the above condition (5.15), we see that τ_1 controls the number of input data points in $S_1^+, S_1^0$and S_1^-. When τ_1 is small, lot many input data points belong to S_1^+ and hence S_1^0 and S_1^- will contain less number of data points which implies the result is sensitive to feature noise around the decision boundary. However, when $\tau_1 = 1$, all the three sets contain many input points and thus the result is less sensitive to noise on $\boldsymbol{x}_i$. Figure

5.1 shows the results of TPMSVM and L1-Pin-NPSVM on a 2-D synthetic dataset where the positive and negative class data points are marked by red triangles and blue circles respectively. Using the results of (5.3) for $\tau_1 = 0.1$ and $\tau_1 = 0.5$, the non-parallel hyperplanes $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0$ and $\boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0$ along with the boundary of the decision hyperplane are illustrated in Figure 5.1(b) and Figure 5.1(c) respectively. In the figures, data points lying below the positive hyperplane $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0$ will be the set S_1^+ (lightly shaded region) and those data points which lie above the hyperplane will form the set S_1^- (non-shaded region). When the value of τ_1 becomes large, the sets $S_1^+, S_1^0$and S_1^-contain many data points and the result is less susceptible to noise.



(a) TPMSVM

(b) L1-Pin-NPSVM (τ = 0.1)

(c) L1-Pin-NPSVM (τ = 0.5)

Figure 5.1 Classification results of linear (a) TPMSVM, (b) L1-Pin-NPSVM with and (c) L1-Pin-NPSVM with on a 2D synthetic dataset. The region corresponding to $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 > 0$ is shown lightly shaded and S_1^+denotes all the positive class data points falling in this region. As the value of τ_1 increases, the margin between the positive $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0$and negative $\boldsymbol{w}_2^t\boldsymbol{x} + b_2 = 0$ hyperplanes become larger and hence less number of points inS_1^+.

Let $\boldsymbol{x}_0$ be a data point such that $\boldsymbol{w}_1^t\boldsymbol{x}_0 + b_1 = -1$ is satisfied. Using the sum of the distance between $\boldsymbol{x}_0$ and negative class samples $\boldsymbol{x}_i \in T$, define the scatter of $\boldsymbol{x}_i$ around $\boldsymbol{x}_0$ as $\sum_{i=m_1+1}^{m_1+m_2} |\boldsymbol{w}_1^t(\boldsymbol{x}_i - \boldsymbol{x}_0)| = \sum_{i=m_1+1}^{m_1+m_2} |\boldsymbol{w}_1^t\boldsymbol{x}_i + b_1 + 1| = ||B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2||_1$. Thus, we can see that minimization of $||B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2||_1$ in (5.3a) can be interpreted as the scatter minimization of negative class data points around $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 = -1$.

Similarly, the scatter of $\boldsymbol{x}_i$ from the positive class around a data point $\widetilde{\boldsymbol{x}}_0$ so that $\boldsymbol{w}_1^t\widetilde{\boldsymbol{x}}_0 + b_1 = 0$ becomes $\sum_{i=1}^{m_1} |\boldsymbol{w}_1^t(\boldsymbol{x}_i - \widetilde{\boldsymbol{x}}_0)| = \sum_{i=1}^{m_1} |0 - y_i(\boldsymbol{w}_1^t\boldsymbol{x}_i + b_1)|$ and therefore its minimization is the scatter minimization of positive class data points around $\boldsymbol{w}_1^t\boldsymbol{x} + b_1 = 0$.

Consider the following problem

$$\min_{\boldsymbol{w}_1, b_1} \frac{1}{2} c_3(||\boldsymbol{w}_1||^2 + b_1^2) + ||B\boldsymbol{w}_1 + b_1\boldsymbol{e}_2 + \boldsymbol{e}_2||_1 + c_5 \sum_{i=1}^{m_1} |0 - y_i(\boldsymbol{w}_1^t\boldsymbol{x}_i + b_1)| \qquad (5.16)$$

Where $c_5 > 0$ is a parameter. Note that the first term is the regularization term and by minimizing it, the structural risk minimization principle is implemented. Adding the hinge loss based misclassification error term $c_7 \sum_{i=1}^{m_1} max\{0,0 - y_i(\boldsymbol{w}_1^t x_i + b_1)\}$ into (5.16), where $c_7 > 0$ is a parameter, the problem formulation (5.3a) is obtained by choosing $c_1 = c_5 + c_7$ and $\tau_1 = \frac{c_5}{c_1}$, i.e., pinball can be regarded as a combination of within class scatter and the hinge loss misclassification error together. From the above derivation, we have $0 \leq \tau_1 \leq 1$. A similar conclusion can be drawn for (5.3b).

5.3 Discussion on L1-Pin-NPSVM

The proposed L1-Pin-NPSVM is compared to TPMSVM, Pin-SVM and Pin-TSVM in this section.

5.3.1 L1-Pin-NPSVM versus TPMSVM

Both TPMSVM and L1-Pin-NPSVM seek two non-parallel hyperplanes and they are obtained by solving a pair of QPPs with box constraints. They allow a unified framework for the linear and non-linear kernels where the kernel trick can be applied directly like in the classical SVM. Both of them lose sparsity. However, L1-Pin-NPSVM is significantly different from TPMSVM in the sense that TPMSVM employs hinge loss whereas pinball loss is used in L1-Pin-NPSVM. Because of pinball loss, correctly classified points are also penalized in L1-Pin-NPSVM. When the size of both classes of data points is equal, the computational complexity of TPMSVM is $2O((m/2)^3)$ whereas for L1-Pin-NPSVM it is $2O(m^3)$. In Figure 5.1, the classification results for a synthetic dataset by TPMSVM and L1-Pin-NPSVM are illustrated. Since TPMSVM employs hinge loss, the margin between the positive and negative hyperplanes is small. On the contrary, the margin between the non-parallel hyperplanes of L1-Pin-NPSVM can be controlled using pinball loss parameters.

5.3.2 L1-Pin-NPSVM versus Pin-SVM

Pinball is employed in both Pin-SVM and L1-Pin-NPSVM. Like Pin-SVM, the proposed L1-Pin-NPSVM also enjoys the benefits of noise insensitivity and stability in terms of re-sampling. Pin-SVM leads to solving a single QPP in *2m* nonnegative variables subject to *(m+1)* equality constraints with computational complexity $O((2m)^3)$ whereas L1-Pin-NPSVM solves two smaller sized QPPs in m variables subject to m box constraints and its computational complexity is $2O(m^3)$. Note that, solving two smaller QPPs instead of a single large QPP makes the learning speed of L1-Pin-NPSVM much faster than Pin-SVM. Lastly, one can observe that Pin-SVM is formulated with a single pinball loss parameter value τ whereas in L1-Pin-NPSVM one can set the pinball loss parameter values τ_1, τ_2 separately corresponding to data points of positive and negative classes. This property makes L1-Pin-NPSVM more suitable for problems with imbalanced data.

5.3.3 L1-Pin-NPSVM versus Pin-TSVM

Both the methods are non-parallel SVM classifiers in the spirit of TWSVM where pinball loss of the form (5.2) is employed for misclassification terms. L1-Pin-NPSVM minimizes the sum of projection values of negative (positive) input data points on the hyperplane $f_1(\boldsymbol{x}) = -1$ ($f_2(\boldsymbol{x}) = 1$) in L_1 −norm whereas in case of Pin-TSVM it is minimized on the positive (negative) hyperplane $f_1(\boldsymbol{x}) = 0$ ($f_2(\boldsymbol{x}) = 0$). If we assume that the size of the datasets of both the classes is approximately equal, i.e. $m_1 \approx m_2 \approx m/2$, Pin-TSVM solves a pair of QPPs having m variables of computational complexity $2O(m^3)$. Both the methods have the following advantages: (i). like in the classical SVM, kernel trick can be applied directly; (ii). the pair of QPPs are free from inverse of kernel matrices; (iii). noise insensitivity, small within-class scatter and small misclassification error properties are satisfied.

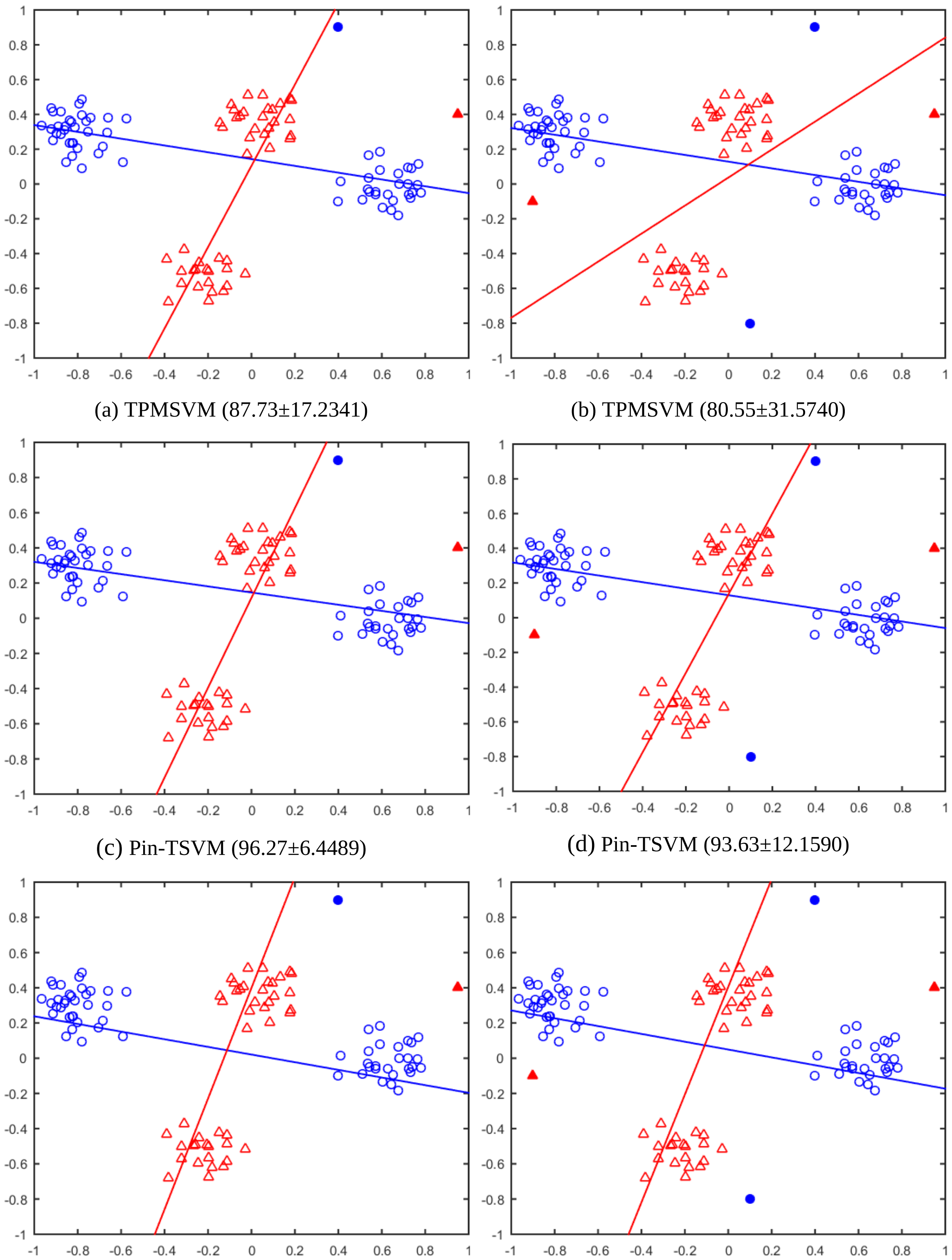
5.4 Experimental Results

In this section, we conduct the performance evaluation of our method L1-Pin-NPSVM by comparing its experimental results with the state-of-the-art SVM methods: TPMSVM, Pin-SVM and Pin-TSVM on two synthetic and 15 benchmark datasets from UCI repository of machine learning datasets (Murphy & Aha, 1992). As synthetic datasets, *Cross-planes* (Mangasarian & Wild) and Ripley (Ripley, 1996) datasets are chosen to test the robustness in the presence of outliers and feature noise. On the other hand, we have polluted the benchmark datasets (Murphy & Aha, 1992) by adding noise with three levels of variance. All the algorithms are implemented in MATLAB R2015a on Windows 10 OS running on a PC configured with 64 bit, 3.40 GHz

Intel®core™ i7 processor having 8 GB of RAM. The MOSEK optimization toolbox for MATLAB available at http://www.mosek.com is used to solve the QPPs. The popular Gaussian kernel function of the form: $k(\boldsymbol{x},\boldsymbol{z}) = exp(-||\boldsymbol{x}-\boldsymbol{z}||^2/2\sigma^2)$ is applied where $\sigma > 0$ is a parameter. It is known that parameter values can influence the performance of the classifier. In this study, the parameters of SVMs are determined by performing tenfold cross-validation strategy where the penalty parameters c's, ν_1, ν_2 and the kernel parameter σ are chosen by varying them in the range of $\{2^i|$ i = -9, -8, … ,9$\}$ whereas τ_1, τ_2 are varied in {0.1, 0.2, 0.5, 1}. For simplicity, we assumed $c_1 = c_2$, $c_3 = c_4$, $\nu_1 = \nu_2$, $\tau_1 = \tau_2$. In our experiments, by: (i) randomly dividing the dataset into 10 equal parts in which nine of them are taken together to be the training set and the remaining one as a test set, the accuracy is computed; (ii) repeating the process 10 times, the average test accuracy is computed and is taken as the classification accuracy.

5.4.1 Cross-planes synthetic dataset

In this sub-section, we perform experiments on a 2D *Cross-planes* dataset consisting of 46 positive data points and 58 negative data points generated by perturbing them originally lying on two intersecting lines belonging in two classes. In Figure 5.2, they are represented by red triangles and blue circles respectively. To test the robustness of L1-Pin-NPSVM, we introduce outliers in the dataset and analyze the changes of the results after adding them. For this purpose, the dataset is contaminated having two outliers in one case and four in the second case (Yan et al., 2018). Along with the input data points and the outliers, the learned linear non-parallel hyperplanes of TPMSVM, Pin-TSVM and L1-Pin-NPSVM are shown in Figure 5.2(a)- Figure 5.2(f) where the outliers are marked by filled red triangles and blue circles. In addition, the classification accuracies and the plus or minus standard deviations are also indicated. All the results are based on the optimal parameters. For the problem of two outliers, the results of classification accuracies of TPMSVM, Pin-TSVM and L1-Pin-NPSVM are 87.73%, 96.27% and 99.09% respectively whereas, in case of four outliers, their accuracies are 80.55%, 93.63% and 98.18% respectively. The degradation in the classification performance of the algorithms confirms their sensitivity to outliers. Among them, TPMSVM shows the worst accuracy and performance degradation. Further, better accuracy and smaller performance degradation from 99.09% to 98.18% by L1-Pin-NPSVM than Pin-TSVM clearly confirm its superiority in suppressing the influence of outliers present in the given dataset.



(a) TPMSVM (87.73±17.2341)

(b) TPMSVM (80.55±31.5740)

(c) Pin-TSVM (96.27±6.4489)

(d) Pin-TSVM (93.63±12.1590)

(e) L1-Pin-NPSVM (99.09±2.8748)

(f) L1-Pin-NPSVM (98.18±5.7946)

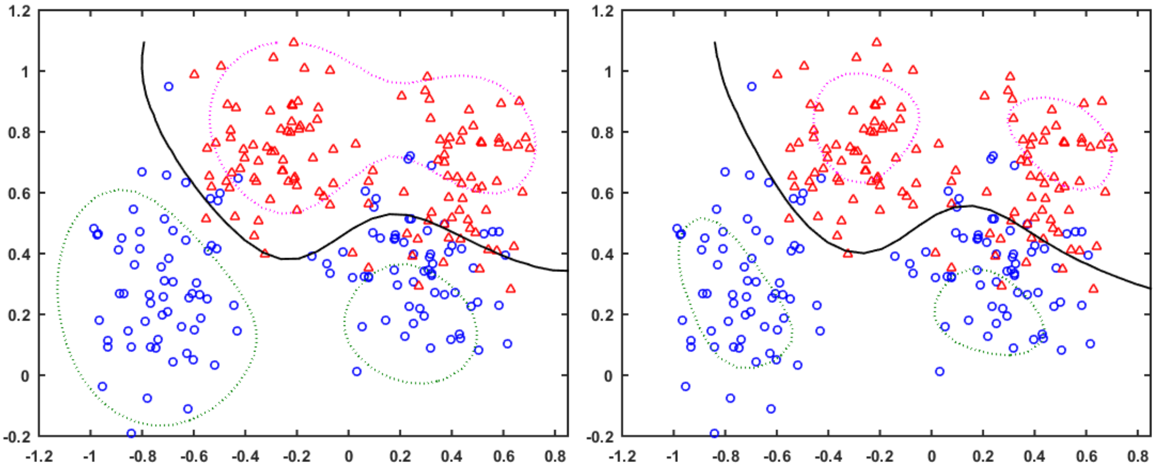
Figure 5.2 Results of nonparallel hyperplanes generated by linear TPMSVM, Pin-TSVM and L1-Pin-NPSVM on "*Cross-planes*" dataset with two outliers (Left) and four outliers (Right). 'Triangles' and 'circles' represent positive and negative class points. Outliers are marked by filled triangles and circles.

In summary, the proposed L1-Pin-NPSVM is superior to Pin-TSVM and TPMSVM in terms of accuracy and robustness which confirms that it is an efficient learning method for classification

5.4.2 **Ripley synthetic dataset**

We test the effectiveness of L1-Pin-NPSVM by considering the Ripley dataset (Ripley, 1996) in R^2 where the positive and negative classes of input points are represented by red triangles and blue circles respectively in Figure 5.3. The dataset consists of 250 training and 1000 test points. The learned classifiers and the nonparallel hyperplanes of Pin-TSVM and L1-Pin-NPSVM for nonlinear kernel are shown in Figure 5.3(a) and Figure 5.3(b). The prediction accuracies for the linear and nonlinear kernels of Pin-TSVM and L1-Pin-NPSVM are tabulated in Table 5.1. Clearly, L1-Pin-NPSVM shows better performance than Pin-TSVM in both cases. According to (Peng et al., 2016), for any test point $\boldsymbol{x}_i \in R^2$ we compute $\boldsymbol{d}_i = (d_1^i, d_2^i)$ where d_1^i and d_2^i are the distances from $\boldsymbol{x}_i$ to the nonparallel hyperplanes $f_1(\boldsymbol{x}) = 0$ and $f_2(\boldsymbol{x}) = 0$ respectively. If $d_1^i < d_2^i$ then we assign the class label (+1) to $\boldsymbol{x}_i$ else its class label will be (-1). For the purpose of illustration, any test point $\boldsymbol{x}_i \in R^2$ is plotted with (d_1^i, d_2^i) as its coordinates.

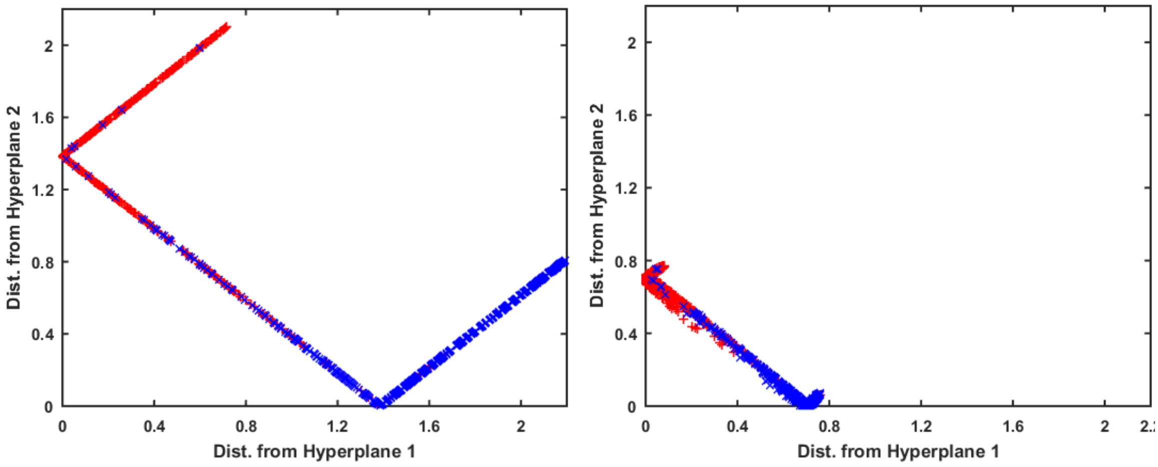
By considering a part of test points, the 2D scatter plots for Pin-TSVM and L1-Pin-NPSVM are illustrated in Figure 5.4 where the horizontal and vertical axes represent the distance of a point from hyperplane1 and hyperplane2 respectively.



(a) Pin-TSVM

(b) L1-Pin-NPSVM

Figure 5.3 Experimental results of Pin-TSVM and L1-Pin-NPSVM on Ripley dataset. The positive and negative classes of data points are marked by ‘triangle’ and ‘circle’. The kernel classifier and the non-parallel hyperplanes are shown as solid and dashed curves.



(a) Pin-TSVM (b) L1-Pin-NPSVM

Figure 5.4 Two dimensional projection of kernel Pin-TSVM and L1-Pin-NPSVM for a part of test data points from *Ripley* dataset.

In both methods, majority of test points are clustered around their respective hyperplanes. For the proposed method, most test points are very close to their respective hyperplanes. We compute the following values for the test points

$$A_1 = \frac{\sum_{y_i=+1} d_1^i}{\sum_{y_i=+1} d_2^i} \text{ and } A_2 = \frac{\sum_{y_i=(-1)} d_1^i}{\sum_{y_i=(-1)} d_2^i}.$$

Note that whenever $A_1$and A_2 are smaller, better accuracy performance is achieved. In addition to their learning accuracies the values of A_1 and A_2 by Pin-TSVM and L1-Pin-NPSVM for both the linear and nonlinear kernels are reported in Table 5.1. Since the ordered pair of values (A_1, A_2) for the nonlinear kernel is much smaller than the linear kernel, we say that nonlinear kernel performs better than linear kernel. Also, smaller value of (A_1, A_2) for L1-Pin-NPSVM than Pin-TSVM indicates the superiority of L1-Pin-NPSVM.

Table 5.1 Experimental results of Pin-TSVM and L1-Pin-NPSVM on Ripley dataset

Type of kernel (Train size, Test size)	Pin-TSVM Accuracy		L1-Pin-NPSVM Accuracy	
	A1	A2	A1	A2
Linear kernel (250X2,1000 X2)	89.20		89.50	
	0.3577	0.3476	0.2887	0.3418
Gaussian kernel (250 X2 X1000 X2)	91.10		92.10	
	0.2500	0.2614	0.2156	0.1625

5.4.3 **Benchmark datasets**

In this subsection, we test the validity of the proposed method for linear and Gaussian kernels by conducting numerical experiments on 15 commonly used benchmark datasets from the UCI machine learning repository (Murphy & Aha, 1992). All the input data points are normalized to lay their feature values in the unit interval [0, 1]. For the purpose of checking noise insensitivity of L1-Pin-NPSVM, both the training and testing data points are distorted by zero-mean Gaussian noise in which the ratio of variance of noise to that of feature represented as r, is taken as $r = 0$ (i.e., noise-free), $r = 0.05$ and $r = 1$ (Huang et al., 2014).

Table 5.2 Performance comparison of the proposed method L1-Pin-NPSVM with TPMSVM, Pin-SVM and Pin-TSVM on real-world benchmark datasets with noise level r. The Linear kernel was used. Training time is measured in seconds.

Dataset (Total size)	Ratio of noise	TPMSVM Time (Rank)	Pin-SVM Time (Rank)	Pin-TSVM Time (Rank)	L1-Pin-NPSVM Time (Rank)
Cleveland (297X 13)	*r*=0	84.18±7.35 0.0467 (3)	84.17±5.49 0.1338 (4)	84.40±4.15 0.0450 (2)	**85.25±6.05** 0.0427 (1)
	r=0.05	83.43±7.28 0.0452 (4)	84.10±5.06 0.1344 (3)	84.25±4.20 0.0428 (2)	**85.05± 6.30** 0.0462 (1)
	r=0.1	83.25±7.22 0.0456 (4)	84.30±4.90 0.1337 (3)	84.40±4.14 0.0477 (2)	**84.51± 6.90** 0.0483 (1)
Breast Cancer (683 X9)	*r*=0	96.65±2.46 0.4368 (4)	96.80±1.73 0.8396 (3)	96.98±2.77 0.6546 (2)	**97.66±1.83** 0.4780 (1)
	r=0.05	96.60±3.62 0.4376 (2)	96.40±1.97 0.8169 (4)	96.55±2.85 0.6511 (3)	**97.52±2.05** 0.4785 (1)
	r=0.1	96.06±3.80 0.44380 (4)	96.10±2.96 0.8159 (2.5)	96.10±2.96 0.6589 (2.5)	**97.37±1.98** 0.4785 (1)
WPBC (194 X 33)	*r*=0	78.95±9.33 0.0216 (4)	79.58±10.12 0.0557 (2)	79.35±10.74 0.0389 (3)	**79.94±9.01** 0.0265 (1)
	r=0.05	78.67±9.30 0.0220 (4)	**79.40±10.27** 0.0550 (1)	79.30±10.81 0.0375 (2.5)	79.30±9.15 0.0305 (2.5)
	r=0.1	77.89±9.26 0.0220 (4)	79.12±10.37 0.0555 (3)	**79.22±10.87** 0.0390 (1)	79.15±9.40 0.0308 (2)
Sonar (208 X 60)	*r*=0	78.91±8.53 0.0308 (2)	77.45±6.08 0.0639 (4)	77.85±5.98 0.0433 (3)	**79.98±6.10** 0.0385 (1)
	r=0.05	78.55±8.78 0.0315 (4)	78.90±5.10 0.0646 (3)	79.27±5.71 0.0406 (2)	**79.85±5.91** 0.0379 (1)
	r=0.1	77.15±8.90 0.0312 (4)	78.85±5.54 0.0643 (3)	79.08±5.90 0.0438 (2)	**79.83±.6.05** 0.0385 (1)
Heart statlog (270 X 13)	*r*=0	84.25±6.34 0.0421 (4)	**84.70±6.65** 0.1087 (1.5)	**84.70±6.65** 0.0626 (1.5)	84.60±6.12 0.0345 (3)
	r=0.05	83.90±6.54	**84.44±7.65**	84.30±7.81	**84.44±6.5**0

		0.0420 (4)	0.1090 (1.5)	0.0533 (3)	0.0361 (1.5)
	r=0.1	82.80±6.99 0.0418 (2)	82.40±9.45 0.1097 (4)	82.59±8.86 0.0516 (3)	**83.33±6.02** 0.0396 (1)
Votes (435 X 16)	*r*=0	94.47±3.16 0.1120 (4)	**96.20±4.31** 0.2516 (1.5)	**96.20±4.31** 0.1245 (1.5)	96.15±3.71 0.1449 (3)
	r=0.05	94.47±3.20 0.1115 (4)	95.90±4.20 0.2502 (3)	**96.05±3.86** 0.1204 (1)	95.95±3.51 0.1791 (2)
	r=0.1	94.24±3.95 0.1120 (4)	95.80±4.32 0.2518 (2)	95.60±2.73 0.1267 (3)	**95.90±3.26** 0.1780 (1)
Haberman (306X 3)	*r*=0	73.47±9.10 0.0710 (3)	73.53±7.42 0.1403 (2)	73.45±6.50 0.1282 (4)	**77.70±7.12** 0.0821 (1)
	r=0.05	73.80±9.97 0.0696 (2)	73.50±7.38 0.1386 (3)	73.30±6.74 0.1246 (4)	**77.20±7.30** 0.0830 (1)
	r=0.1	73.00±9.10 0.0705 (4)	73.10±7.12 0.1390 (2.5)	73.10±7.12 0.1236 (2.5)	**76.09±7.31** 0.0855 (1)
Ionosphere (351 X 33)	*r*=0	89.78±3.67 0.0890 (4)	91.60±3.62 0.1983 (3)	91.73±2.73 0.0995 (2)	**91.80±3.65** 0.0845 (1)
	r=0.05	88.20±3.95 0.0891 (4)	90.40±3.95 0.1972 (3)	90.45±.3.90 0.0881 (2)	**91.45±3.42** 0.0857 (1)
	r=0.1	86.35±4.11 0.0890 (4)	88.54±5.12 0.1954 (3)	88.70±3.98 0.0975 (2)	**89.25±3.59** 0.0870 (1)
Spect (267 X 22)	*r*=0	**79.40±5.39** 0.0390 (1)	78.28±5.85 0.1325 (4)	78.35±7.99 0.0463 (3)	78.90±5.04 0.0395 (2)
	r=0.05	78.60±5.25 0.0412 (3.5)	78.90±5.26 0.1364 (2)	**78.95±6.23** 0.0482 (1)	78.60±5.39 0.0390 (3.5)
	r=0.1	78.40±5.21 0.0405 (3.5)	**78.90±5.18** 0.1351 (1)	78.60±4.35 0.0481 (2)	78.40±5.35 0.0406 (3.5)
Heart-C (303 X13)	*r*=0	**83.50±5.88** 0.0491 (1)	83.10±7.09 0.1459 (4)	83.21±7.12 0.0480 (3)	83.45±6.46 0.0456 (2)
	r=0.05	82.49±8.08 0.0485 (3)	82.20±6.77 0.1472 (4)	83.50±6.80 0.0492 (2)	**83.69±6.55** 0.0470 (1)
	r=0.1	81.84±6.65 0.0490 (3)	81.54±7.31 0.1458 (4)	83.17±6.47 0.0490 (2)	**83.60±5.51** 0.0484 (1)
WDBC (569 X 30)	*r*=0	96.42±4.97 0.3812 (4)	97.50±2.20 0.6589 (2)	**97.68±2.64** 0.4950 (1)	97.25±1.74 0.3725 (3)
	r=0.05	96.30±4.86 0.3805 (4)	**97.25±1.90** 0.6514 (1.5)	**97.25±1.86** 0.4947 (1.5)	97.18±2.01 0.3720 (3)
	r=0.1	96.08±4.89	97.10±1.99	**97.20±1.10**	96.90±1.43

		0.3810 (4)	0.6524 (2)	0.4809 (1)	0.3890 (3)
Transfusion (748 X 4)	r=0	77.30±7.01 0.5892 (4)	**78.25±4.15** 0.9770 (1)	77.98±4.24 0.7895 (3)	78.10±5.13 0.5608 (2)
	r=0.05	76.70±6.91 0.5873 (4)	**77.80±4.32** 0.9722 (1)	77.50±4.62 0.7757 (3)	77.60±5.40 0.5801 (2)
	r=0.1	76.41±6.98 0.5890 (4)	76.55±4.85 0.9751 (2)	76.50±4.80 0.7896 (3)	**77.03±5.60** 0.5886 (1)
Pima Indians Diabetes (768 X 8)	r=0	75.91±6.00 0.6084 (4)	76.95±5.78 1.1805 (3)	77.20±4.08 1.0090 (2)	**77.61±6.16** 0.7186 (1)
	r=0.05	75.58±6.10 0.6048 (4)	76.78±6.87 1.1721 (2)	76.70±6.54 0.9865 (3)	**77.16±6.40** 0.7105 (1)
	r=0.1	75.89±6.48 0.6080 (4)	76.15±6.42 1.1782 (3)	76.35±569 1.0098 (2)	**77.03±5.62** 0.7093 (1)
German (1000 X 24)	r=0	76.16±7.22 0.7253 (3)	**76.70±3.16** 1.8821 (1)	76.10±3.89 1.2623 (4)	76.21±5.20 1.1531 (2)
	r=0.05	74.98±7.43 0.7498 (4)	75.45±4.91 1.9518 (3)	75.50±4.85 1.3860 (2)	**76.04±5.25** 1.1601 (1)
	r=0.1	73.91±8.01 0.7438 (4)	75.30±5.86 1.8963 (2.5)	75.30±5.80 1.2891 (2.5)	**75.60±5.40** 1.1583 (1)
QSAR (1055 X 41)	r=0	85.12±9.58 0.7586 (4)	87.39±3.25 2.5416 (3)	87.98±4.01 1.4010 (2)	**88.90±6.50** 1.3128 (1)
	r=0.05	84.69±9.42 0.7694 (4)	85.20±3.45 2.7536 (3)	86.45±4.76 1.2856 (2)	**87.52±6.72** 1.1064 (1)
	r=0.1	82.98±9.86 0.7580 (4)	84.54±3.78 2.7311 (3)	84.65±5.42 1.3951 (2)	**85.16±6.86** 1.1180 (1)
Average rank		3.5556	2.6111	2.3	1.5333

Table 5.3 Average ranks on the accuracy by SVMs for linear kernel on real-world datasets with noise level r

Noise	TPMSVM	Pin-SVM	Pin-TSVM	L1-Pin-NPSVM
r=0	3.2667	2.6	2.4667	1.6667
r=0.05	3.6333	2.5333	2.2667	1.5667
r=0.1	3.7667	2.7	2.1667	1.3667

Table 5.4 Performance comparison of the proposed method L1-Pin-NPSVM with TPMSVM, Pin-SVM, and Pin-TSVM on real-world benchmark datasets with noise level r. Gaussian kernel was used. Training time is measured in seconds.

Dataset (Total size)	Ratio of noise	TPMSVM Time (Rank)	Pin-SVM Time (Rank)	Pin-TSVM Time (Rank)	L1-Pin-NPSVM Time (Rank)
Cleveland (297X 13)	*r*=0	84.18±7.36 0.0786 (4)	84.25±8.55 0.1428 (3)	85.18±6.22 0.0866 (2)	**85.49±6.30** 0.0910 (1)
	r=0.05	83.84±6.82 0.0798 (4)	84.34±6.57 0.1444 (2)	84.30±6.45 0.0871 (3)	**85.48±6.86** **0.0930** (1)
	r=0.1	83.81±8.21 0.0790 (4)	84.88±5.97 0.1439 (2)	84.79±6.07 0.0893 (3)	**85.08±6.82** **0.0920** (1)
Breast Cancer (683 X9)	*r*=0	97.18±1.42 0.6289 (4)	97.42±0.83 1.1370 (3)	**97.95±1.57** 0.7906 (1.5)	**97.95±1.62** 0.6918 (1.5)
	r=0.05	97.15±1.48 0.6086 (3)	97.06±1.95 1.1362 (4)	97.80±1.58 0.7897 (2)	**97.90±1.65** 0.6996 (1)
	r=0.1	97.30±1.95 0.6349 (4)	97.36±1.93 1.1374 (3)	**97.80±1.58** 0.7917 (1.5)	**97.80±1.67** 0.6914 (1.5)
WPBC (194 X 33)	*r*=0	79.80±6.76 0.0462 (4)	82.17±9.88 0.0767 (2)	81.75±7.88 0.0495 (3)	**82.52±9.63** 0.0498 (1)
	r=0.05	78.32±6.89 0.0428 (4)	**81.80±9.81** 0.0747 (1)	80.95±6.81 0.0489 (3)	81.79±9.40 0.0430 (2)
	r=0.1	77.78±5.47 0.0428 (4)	80.44±5.81 0.0704 (3)	80.50±5.60 0.0489 (2)	**81.05±9.21** 0.0425 (1)
Sonar (208 X 60)	*r*=0	88.02±7.14 0.0590 (4)	90.85±5.30 0.0740 (2)	90.80±5.30 0.0622 (3)	**92.33±5.09** 0.0481 (1)
	r=0.05	87.52±7.20 0.0585 (4)	89.62±6.30 0.0749 (2)	89.60±6.30 0.0617 (3)	**91.35±6.20** 0.0470 (1)
	r=0.1	86.59±7.14 0.0596 (4)	88.58±6.70 0.0741 (2)	88.50±6.70 0.0621 (3)	**90.35±.6.48** 0.0491 (1)
Heart statlog (270 X 13)	*r*=0	**85.55±3.68** 0.0795 (1)	84.80±9.40 0.1168 (4)	84.81±4.93 0.0890 (2.5)	84.81±5.90 0.0840 (2.5)
	r=0.05	84.07±4.29 0.0806 (4)	84.55±10.39 0.1180 (2)	**84.70±5.9**0 0.0865 (1)	84.50±5.75 0.0845 (3)
	r=0.1	83.09±5.96 0.0820 (4)	83.54±5.46 0.1165 (2)	**83.60±5.56** 0.0878 (1)	83.50±5.73 0.0863 (3)
Votes	*r*=0	94.93±2.91 0.1391 (4)	**96.50±3.91** 0.2915 (1.5)	96.32±2.98 0.1434 (3)	**96.50±2.87** 0.2153 (1.5)

(435 X 16)	*r*=0.05	94.70±3.44 0.1384 (4)	**96.50±3.07** 0.2900 (1)	96.20±2.07 0.1409 (3)	96.32±2.88 0.2354 (2)
	r=0.1	94.16±2.94 0.1370 (4)	96.18±1.55 0.2869 (2)	96.10±1.86 0.1406 (3)	**96.30±2.85** 0.2301 (1)
Haberman (306X 3)	*r*=0	**76.73±8.20** 0.0950 (1)	75.54±9.07 0.2552 (4)	75.81±7.54 0.1491 (3)	76.37±7.63 0.0941 (2)
	r=0.05	75.75±8.28 0.0974 (2)	75.17±7.41 0.2544 (3.5)	75.17±7.43 0.1442 (3.5)	**76.40±7.36** 0.0928 (1)
	r=0.1	74.79±8.51 0.0993 (2)	74.20±6.54 0.2513 (3.5)	74.20±6.54 0.1417 (3.5)	**76.07±7.39** 0.0978 (1)
Ionosphere (351 X 33)	*r*=0	94.02±4.02 0.1158 (4)	94.30±2.33 0.2265 (3)	94.59±1.33 0.1218 (2)	**95.45±4.05** 0.1694 (1)
	r=0.05	92.61±4.10 0.1160 (3)	92.87±2.03 0.2206 (2)	92.30±1.20 0.1159 (4)	**94.03±3.86** 0.1701 (1)
	r=0.1	90.97±4.86 0.1170 (4)	91.46±5.85 0.2148 (2)	91.29±2.96 0.1270 (3)	**94.03±3.80** 0.1718 (1)
Spect (267 X 22)	*r*=0	81.65±4.47 0.0780 (4)	82.05±6.98 0.1396 (3)	82.06±3.68 0.0854 (2)	**83.30±5.60** 0.802 (1)
	r=0.05	80.48±4.88 0.0792 (3)	80.40±5.72 0.1421 (4)	82.50±3.43 0.0874 (2)	**83.16±5.43** 0.0794 (1)
	r=0.1	80.10±5.00 0.0785 (4)	80.16±6.41 0.1408 (3)	81.30±3.70 0.0892 (2)	**82.08±5.89** 0.0790 (1)
Heart-C (303 X13)	*r*=0	**85.51±5.35** 0.0868 (1.5)	84.81±5.26 0.1783 (4)	85.12±3.96 0.1021 (3)	**85.51±4.11** 0.0960 (1.5)
	r=0.05	83.82±6.18 0.0832 (4)	84.65±5.65 0.1795 (2)	**84.80±4.19** 0.1038 (1)	84.60±4.28 0.0976 (3)
	r=0.1	83.82±6.15 0.0883 (4)	83.98±5.60 0.1787 (2)	83.90±4.24 0.1034 (3)	**84.05±4.30** 0.0960 (1)
WDBC (569 X 30)	*r*=0	97.18±2.13 0.5164 (4)	**98.08±2.45** 1.0744 (1)	97.98±3.78 0.5484 (3)	98.00±1.16 0.3814 (2)
	r=0.05	96.67±2.01 0.4856 (4)	97.78±1.69 1.1068 (3)	**97.86±3.69** 0.5130 (1)	97.84±1.18 0.4010 (2)
	r=0.1	96.66±1.87 0.4860 (4)	97.20±1.86 1.0717 (3)	**97.80±3.8**6 0.4923 (1.5)	**97.80±1.20** 0.4187 (1.5)
Transfusion (748 X 4)	*r*=0	78.77±2.85 0.7676 (4)	79.70±2.74 2.4669 (2.5)	**80.10±2.80** 0.9870 (1)	79.70±2.03 0.8975 (2.5)
	r=0.05	76.63±4.18 0.7410 (4)	78.65±5.26 2.4828 (3)	79.01±3.36 0.9725 (2)	**79.40±2.25** 0.8143 (1)

	r=0.1	76.36±4.12 0.7403 (4)	77.25±3.96 2.4632 (3)	77.75±4.60 0.9246 (2)	**78.15±2.67** 0.8190 (1)
Pima Indians Diabetes (768 X 8)	r=0	77.3±5.98 0.5135 (4)	78.52±4.54 2.8370 (2)	78.50±3.56 1.0114 (3)	**78.65±5.91** 0.9103 (1)
	r=0.05	76.80±5.97 0.5145 (4)	77.60±6.06 2.4607 (2)	77.50±5.99 1.0148 (3)	**78.14±6.20** 0.9873 (1)
	r=0.1	76.35±5.31 0.5196 (4)	77.12±4.34 2.3969 (2)	77.09±4.69 1.0135 (3)	**77.61±6.96** 0.9807 (1)
German (1000 X 24)	r=0	76.90±5.54 1.0861 (4)	**77.30±5.59** 4.1229 (1)	77.10±1.86 2.5687 (3)	77.25±5.10 2.0153 (2)
	r=0.05	75.25±5.60 1.0250 (4)	76.30±5.59 4.2436 (3)	76.70±1.85 2.6473 (2)	**76.80±5.20** 1.9729 (1)
	r=0.1	75.16±5.98 1.0982 (4)	76.00±5.86 4.4651 (3)	76.10±1.92 2.5989 (2)	**76.80±5.21** 1.9631 (1)
QSAR (1055 X 41)	r=0	86.97±6.65 1.1453 (4)	88.43±2.09 5.2027 (3)	89.38±9.92 2.7243 (2)	**90.90±9.66** 2.0948 (1)
	r=0.05	85.98±6.77 1.2328 (4)	88.06±2.77 5.2160 (3)	89.28±9.48 2.7560 (2)	**90.08±9.35** 2.0655 (1)
	r=0.1	85.80±7.79 1.1062 (4)	87.73±3.49 5.2089 (3)	88.34±8.33 2.6815 (2)	**89.21±9.10** 2.0614 (1)
Average rank		3.6556	2.5556	2.4	1.3889

Table 5.5 Average ranks on the accuracy by SVMs for Gaussian kernel on real-world datasets with noise level r

Noise	TPMSVM	Pin-SVM	Pin-TSVM	L1-Pin-NPSVM
r=0	3.4333	2.6	2.4667	1.5
r=0.05	3.6667	2.5	2.3667	1.4667
r=0.1	3.8667	2.5667	2.3667	1.2

As experimental results, the classification accuracy and the learning time based on optimal parameters for the linear and Gaussian kernels are summarized in Table 5.2 and 5.4. The best classification accuracy is boldfaced. We ranked the algorithms in accordance with the classification accuracy attained on each dataset and reported their average ranks. Note that smaller average rank means better prediction ability. In Table 5.2 for the linear kernel, the overall minimum average rank 1.5333 by the proposed L1-Pin-NPSVM indicates its superiority. Among the methods considered for experimental study, L1-Pin-NPSVM achieves the best accuracy at 7 times out of 15 datasets considered. Notice that the other two pinball based classifiers Pin-SVM and Pin-TSVM show comparative performance as they yield the best classification accuracy in equal number of times. Again, out of 45 cases (15 datasets x 3 noise levels), L1-Pin-NPSVM yields the best accuracy in 30 cases. On learning time, it also compares well with SVM. From Table 5.3 on the average ranks for linear kernel, we can see that pinball based SVMs always show better accuracy performance than the hinge loss based TPSVM. At each level of noise, the best average rank is achieved by L1-Pin-NPSVM. In Table 5.4, the comparison results for Gaussian kernel are illustrated. Our L1-Pin-NPSVM formulation achieved the best accuracy on 8 datasets. Also, out of 45 experiments, the best accuracy was shown at 39 times by L1-Pin-NPSVM. Interestingly when $r = 0.05$ and $r = 0.1$, all the three pinball based SVMs show, in general, better classification accuracy than the hinge loss based TPMSVM.

From the results of Table 5.2-5.5, we have the following observations: (i). for all the methods, in general, better accuracy results are achieved for the Gaussian kernel than the linear kernel with extra learning cost; (ii). as the noise level rises, the decrease in the accuracy of each method indicates the effect of noise present in the data; (iii). the lowest average rank by L1-Pin-NPSVM at all levels of noise for both the linear and Gaussian kernels indicate its superiority and robustness; (iv).with the increase in the level of noise, decrease in the average rank values for L1-Pin-NPSVM and whereas the reverse phenomena for TPSVM show the model robustness of L1-Pin-NPSVM to noise in the data which can be due to the use of L_1 −norm; (v). higher learning cost by Pin-SVM is attributed to solving a large sized QPP. From the above observations on the results of comparison, we draw the conclusion that L1-Pin-NPSVM is a superior, robust method for noise and outliers.

To examine the efficiency performance of L1-Pin-NPSVM statistically, we calculate its statistical significance using the Friedman test followed by Nemenyi post-hoc test which is considered to be a simple, safe and robust non-parametric test (Demsar, 2006). We make the accuracy comparison of the algorithms on UCI datasets corrupted by different levels of noise whose average ranks for Gaussian kernel are displayed in Table 5.5. We perform the statistical test for $r = 0$, $r = 0.05$ and $r = 0.1$ in that order.

Under the null hypothesis that all the algorithms are equivalent, by taking $k =$ the number of algorithms, $N =$ the number of datasets and $R_j =$ the average rank of the j^th^ algorithm, we compute the Friedman statistic distributed according to $\chi_F^2 = \frac{12N}{k(k+1)}[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}]$ distribution with $(k-1)$ degrees of freedom and a better statistic $F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$, distributed according to F −distribution with $((k-1), (k-1)(N-1))$ degrees of freedom. Once the null hypothesis is rejected, we proceed with the Nemenyi post-hoc test and perform pair-wise comparison of the algorithms by computing the critical difference. For details, see (Demsar, 2006).

From Table 5.5, we perform the Friedman test and the Nemenyi post-hoc test on the average ranks, assuming the null hypothesis that all four methods are similar. for *r*=0 with *k=4* and *N=15*:

$$\chi_F^2 = \frac{12 \times 15}{4 \times 5}(3.4333^2 + 2.6^2 + 2.4667^2 + 1.5^2 - \frac{4 \times 5^2}{4}) \approx 16.9394,$$

$F_F = \frac{14 \times 16.9394}{15 \times 3 - 16.9394} \approx 8.4514$, where F_F is distributed according to F distribution with $(3, 3 \times 14) = (3,42)$ degrees of freedom. Since the critical value of $F(3,42)$ at the level of significance $\alpha = 0.05$ is 2.8270 and is such that $F_F > 2.8270$, we reject the null hypothesis. Subsequently, we compare the four algorithms in pairs using Nemenyi post-hoc test. Accordingly, since $q_{\alpha=0.10} = 2.291$ (Demsar, 2006), the critical difference (CD) at $p = 0.10$ is $2.291\sqrt{\frac{4 \times 5}{6 \times 15}} \approx 1.08$. Further, if the average ranks of two algorithms differ at least by CD then their performances differ significantly. Thus, from Table 5.5, we derive the pair-wise comparison of L1-Pin-NPSVM with the remaining algorithms by computing the difference between their average ranks. For *r*=0, the difference between the average ranks of : (i). L1-Pin-NPSVM and Pin-TSVM is $2.4667 - 1.5 = 0.9667 < 1.08(CD)$, we say that the post-hoc test is not powerful enough to detect any significant differences between the algorithms; (ii). L1-Pin-NPSVM and the best of Pin-SVM and TPMSVM is $2.6 - 1.5 = 1.1 > 1.08$, implies the performance of L1-Pin-NPSVM is better than Pin-SVM and TPMSVM.

Similarly, to compare the significant performance difference of the algorithms statistically on noise corrupted datasets with *r* = 0.05, we calculate $\chi_F^2 = 22.0245$ and $F_F = 13.4205$. Here, we see that $F_F > F(3,42)$ and hence we reject the null hypothesis. So we perform Nemenyi post-hoc test. From Table 5.5, the difference between the average ranks of: (i). the best and the worst among L1-Pin-NPSVM, Pin-TSVM and Pin-SVM are $2.5 - 1.4667 = 1.0333 < 1.08$, we say that the post-hoc test is not powerful enough to detect any significant differences between the algorithms; (ii). TPMSVM and the worst of Pin-SVM, Pin-TSVM and L1-Pin-NPSVM is $3.6667 - 2.5 = 1.1667 > 1.08$ implies that the performance of TPSVM is inferior to the rest of the algorithms.

Finally, as a statistical comparison of algorithms, using the average ranks for datasets corrupted with noise level *r* = 0.1, we compute $\chi_F^2 = 32.2253$ and $F_F = 35.3162$. Since $F_F > F(3,42)$, we reject the null hypothesis and therefore we proceed with Nemenyi post-hoc test for pair-wise comparison of algorithms. From Table 5.5, the difference between the average ranks of: L1-Pin-NPSVM and the best of the remaining algorithms is 2.3667-1.2 = 1.1667 > 1.08 implies that L1-Pin-NPSVM performs better than the remaining algorithms.

From Table 5.5 and the statistical tests considered, we observe that (i). at all levels of noise, the least average rank is reported by L1-Pin-NPSVM; (ii). when r = 0.1, i.e. for the highest level of noise considered, L1-Pin-NPSVM is the best performer among all the algorithms.

In summary, we conclude that the proposed L1-Pin-NPSVM is an efficient method showing enhanced classification performance and robustness in the presence of noise and outliers.

5.5 Conclusions

With the aim of having the integrated merits of L_1 −norm and pinball loss in achieving enhanced robustness to outliers and bringing noise insensitivity to feature noise, we presented a novel, efficient L_1 −norm based nonparallel support vector machine classifier with pinball loss (L1-Pin-NPSVM) where its associated optimization problem minimizes the scatter loss and the misclassification error by L_1 −norm and pinball loss respectively. The dual formulation of the proposed method solves a pair of QPPs free of inverse kernel matrices. Our formulation allows a unified framework for the linear and nonlinear kernels. The effectiveness of L1-Pin-NPSVM is evaluated on synthetic and UCI benchmark datasets having outliers and/or contaminated by noise. The results confirm its superiority in terms of robustness to outliers and noise insensitivity. In summary, we can conclude that the proposed L1-Pin-NPSVM is an efficient method than the other machine-learning methods for real-world problems when robustness and noise insensitivity is of major concern. Even though L1-Pin-NPSVM is a simple, efficient learning method, it loses sparsity. As future work, an efficient method of solving L1-Pin-NPSVM for problems with large samples will be explored. Possibility of the extension of the proposed method for multi-category classification will be interesting and it will be investigated as another future work.

Chapter 6

L1-norm Support Vector Regression in Primal Based on Huber Loss Function

6.1 Introduction

Support vector regression (SVR) (Cristianini & Shawe-Taylor, 2000; Vapnik, 2000) method becomes the state of the art machine learning tool for data regression because of its excellent generalization performance on many real-world problems. It is well-known that the standard SVR determines the regressor using a predefined epsilon tube around the data points in which the points lying outside the tube contribute to the errors whereas the inside points are simply ignored. To measure the data misfit as stated, the epsilon insensitive function is introduced as a loss function. Construction of robust regression models for noisy data samples or data samples having outliers is a challenging research problem. Loss function plays an important role in obtaining a robust regression model. The popular loss functions used in the literature are the (i). quadratic; (ii). absolute value; (iii). ε-insensitive functions. The quadratic function is smooth but is sensitive to samples having large error of deviation. In comparison with the popular quadratic loss function, absolute value and ε-insensitive functions are less sensitive to noise but they are only continuous and therefore numerical minimization is difficult. However, as a combination of robust treatment to large errors and showing quadratic treatment to small errors, Huber function (Huber & Ronchetti, 2009) was used in the literature to measure the data misfit having the smooth property that it is differentiable everywhere.

In this chapter, we propose a novel robust Huber SVR (HSVR) formulation in primal where the regressor is made as flat as possible by introducing the regularization term in L1-norm. Since the regularization term is non-smooth, it is proposed to replace it by smooth approximation functions and solve the problems by functional iterative method. Tests with both synthetic and real-world data sets confirm the suitability and effectiveness of the proposed robust model.

This chapter is organized as follows. In section 6.2, a novel Huber SVR problem formulation is proposed whose solution is obtained by functional iterative method. The effectiveness of the proposed HSVR in comparison with SVR and LS-SVR is studied in section 6.3. Finally, conclusions are drawn in section 6.4.

6.2 Proposed robust Huber SVR model via 1-norm regularization

Like the ε-insensitive function of Vapnik (Suykens et al., 2002) used in SVR, the Huber M-estimator function (Guitton & Symes, 2003) is convex and is insensitive to noise present in the data, i.e. robust to large error of misfit, which further shows good generalization property on many real-world problems of interest (Camps-Valls et. al., 2006; Gretton et. al., 2001). In this study, we present a regression model whose loss function is the Huber M-estimator function. The proposed Huber SVR model leads to solving an unconstrained minimization problem in primal whose solution is obtained by finding its critical point.

Consider the Huber function asymmetric with respect to γ defined as (Balasundaram & Meena, 2018; Zhu et al., 2008)

$$L_H(x) = \begin{cases} -\gamma_L(2x+\gamma_L) & for -\infty < x < -\gamma_L \\ x^2 & for -\gamma_L \leq x < 0 \\ x^2 & for\ 0 \leq x < \gamma_R \\ \gamma_R(2x-\gamma_R) & for\ \gamma_R \leq x < \infty, \end{cases} \tag{6.1}$$

where $\gamma_L, \gamma_R > 0$ are constants. Clearly, $L_H(\cdot)$ is convex and it changes from quadratic to linear form at $x = -\gamma_L$ and $x = \gamma_R$, and , i.e. it is a hybrid function in which quadratic loss is assigned for small errors and linear otherwise. When $\gamma_L = \gamma_R = \gamma$ in (6.1), becomes the Huber M-estimator (Huber & Ronchetti, 2009)

$$L_H(x) = \begin{cases} x^2 & if|x| \leq \gamma \\ \gamma(2|x|-\gamma) & if|x| > \gamma \end{cases} \tag{6.2}$$

It can be easily verified that the asymmetric Huber function (6.1) can be rewritten as

$$L_H(x) = x^2 - max\{\,(x-\gamma_R),0\}^2 - max\{\,(-x-\gamma_L),0\}^2 \tag{6.3}$$

or equivalently we have

$$L_H(x) = x^2 - (x-\gamma_R)_+^2 - (-x-\gamma_L)_+^2, \tag{6.4}$$

In this work, the equivalent form (6.4) is used for formulating our proposed Huber regression model with 1-norm regularization term and we solve it by functional iterative method.

By calculating the error of misfit via Huber function, the regression function of $f(\boldsymbol{x}) = \boldsymbol{K}(\mathbf{x}, A^t)\mathbf{w} + b$ is obtained by solving the minimization problem

$$\min_{\boldsymbol{u}\in R^{m+1}} ||\boldsymbol{u}||_1 + \frac{C}{2}\sum_{i=1}^{m} L_H(y_i - f(\boldsymbol{x}_i)), \tag{6.5}$$

where $\boldsymbol{u} = \begin{bmatrix} \boldsymbol{w} \\ b \end{bmatrix} \in R^{m+1}$is the unknown vector and $C > 0$ is the regularization constant. However, the identity $|z| = 2z_+ - z$ holds for any $z \in R$ implies $||\boldsymbol{u}||_1 = \boldsymbol{e}_1^t(2\boldsymbol{u}_+ - \boldsymbol{u})$ will be satisfied where $\boldsymbol{e}_1$ is the column vector of ones of size (m+1). Using this result, the Huber SVR in primal (6.5) (HSVR) can be rewritten as

$$\min_{\boldsymbol{u}\in R^{m+1}} L(\boldsymbol{u}) = \boldsymbol{e}_1^t(2\boldsymbol{u}_+ - \boldsymbol{u}) + \frac{C}{2}[||\boldsymbol{y} - G\boldsymbol{u}||^2 - ||(\boldsymbol{y} - G\boldsymbol{u} - \gamma_R\boldsymbol{e})_+||^2 - ||(G\boldsymbol{u} - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+||^2] \quad (6.6)$$

where $G = [K(A, A^t)\boldsymbol{e}]_{m\times(m+1)}$

Remark 6.1. When $\gamma_L = \gamma_R = \gamma$, i.e. for the Huber M-estimator (6.2), the problem (6.6) becomes

$$\min_{\boldsymbol{u}\in R^{m+1}} \boldsymbol{e}_1^t(2\boldsymbol{u}_+ - \boldsymbol{u}) + \frac{C}{2}[||\boldsymbol{y} - G\boldsymbol{u}||^2 - ||(\boldsymbol{y} - G\boldsymbol{u} - \gamma\boldsymbol{e})_+||^2 - ||(G\boldsymbol{u} - \boldsymbol{y} - \gamma\boldsymbol{e})_+||^2].$$

Remark 6.2. The objective function $L(\boldsymbol{u})$ is strongly convex and therefore the minimization problem (6.6) will have a unique minimal solution.

Clearly, $L(\boldsymbol{u})$ is only continuous and hence its gradient does not exist. To overcome the difficulty of the non-differentiability of $L(\boldsymbol{u})$, two approaches are taken in this work: (a). replace the non-smooth vector $\boldsymbol{u}_+$ in the regularization term $\boldsymbol{e}_1^t(2\boldsymbol{u}_+ - \boldsymbol{u})$ by two different smooth approximation functions; (b). application of a generalized derivative for $\boldsymbol{u}_+$. In both approaches, the critical points will be computed by equating their gradients to zero.

Solving non-smooth mathematical programming problems, arising in machine learning methods, using smooth technique is very popular in the literature (Huber & Ronchetti, 2009; Lee & Mangasarian, 2001). In this work, it is proposed to approximate the 'plus' function appearing in the regularization term by smooth functions in two ways leading to two algorithms.

Consider the smooth function used in (Huber & Ronchetti, 2009) for approximating the 'plus' function: for $x \in R$ and $\alpha > 0$, $p_1(x, \alpha) = x + \frac{1}{\alpha}\log(1 + \exp(-\alpha x))$. Taking $x_+ \approx p_1(x, \alpha)$ and denoting $p_1(\boldsymbol{u}, \alpha) = (p_1(u_1, \alpha), \ldots, p_1(u_{m+1}, \alpha))^t$ where $\boldsymbol{u} = (u_1, \ldots, u_{m+1})^t$, problem (6.6) becomes

$$\min_{\boldsymbol{u}\in R^{m+1}} \tilde{L}(\boldsymbol{u}) = \boldsymbol{e}_1^t(2p_1(\boldsymbol{u}, \alpha) - \boldsymbol{u}) + \frac{C}{2}[||\boldsymbol{y} - G\boldsymbol{u}||^2 - ||(\boldsymbol{y} - G\boldsymbol{u} - \gamma_R\boldsymbol{e})_+||^2 - ||(G\boldsymbol{u} - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+||^2] \quad (6.7)$$

By defining $\frac{1}{(e_1 + exp(-\alpha \boldsymbol{u}))} = \left(\frac{1}{1+exp(-\alpha u_1)}, \ldots, \frac{1}{1+exp(-\alpha u_{m+1})}\right)^t$, the critical point for $\tilde{L}(\boldsymbol{u})$ can be computed by solving the root finding problem

$$G^tG\boldsymbol{u} = \frac{\boldsymbol{e}_1}{C} - \frac{2}{C}\left(\frac{1}{\boldsymbol{e}_1 + exp(-\alpha\boldsymbol{u})}\right) + G^t[\boldsymbol{y} + (G\boldsymbol{u} - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u} - \gamma_R\boldsymbol{e})_+] \quad (6.8)$$

This leads to the iterative method of solving the smooth HSVR problem (6.8) (SHSVR1): for i=0,1...

$$\boldsymbol{u}^{i+1} = (G^tG)^{-1}[\frac{\boldsymbol{e}_1}{C} - \frac{2}{C}\left(\frac{1}{\boldsymbol{e}_1 + exp(-\alpha \boldsymbol{u}^i)}\right)$$
$$+G^t[\boldsymbol{y} + (G\boldsymbol{u}^i - \boldsymbol{y} - \gamma_L \boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u}^i - \gamma_R \boldsymbol{e})_+]]$$

Remark 6.3. Following the work of (Huber & Ronchetti, 2009), numerical experiments using SHSVR1 were performed in section 6.3 by taking $\alpha = 5$.

As another smooth approximation of the 'plus' function, consider the quadratic function $p_2(x, x_0) = \frac{x^2}{4|x_0|} + \frac{x}{2} + \frac{|x_0|}{4}$ introduced in (Lee, 2001) where $x, x_0 (\neq 0) \in R$. Clearly,

$$p_2(x, x_0) = x_+ \text{ whenever } |x| = |x_0| \neq 0 \tag{6.9}$$

Replacing x_+ by $p_2(x, x_0)$, problem (6.6) becomes

$$\min_{\boldsymbol{u} \in R^{m+1}} \tilde{\tilde{L}}(\boldsymbol{u}) = 2\boldsymbol{e}_1^t p_2(\boldsymbol{u}, \boldsymbol{u}_0)$$
$$+\frac{C}{2}[\|\boldsymbol{y} - G\boldsymbol{u}\|^2 - \|(\boldsymbol{y} - G\boldsymbol{u} - \gamma_R \boldsymbol{e})_+\|^2 - \|(G\boldsymbol{u} - \boldsymbol{y} - \gamma_L \boldsymbol{e})_+\|^2] \tag{6.10}$$

where $p_2(\boldsymbol{u}, \boldsymbol{u}_0) = (p_2(u_1, u_{01}), \ldots, p_2(u_{m+1}, u_{0,m+1}))^t$ and $\boldsymbol{u}_0 = (u_{01}, \ldots, u_{0,m+1})^t$ is a non-zero vector in R^{m+1}. By equating the gradient of $\tilde{\tilde{L}}(\boldsymbol{u})$ to zero, we get

$$\left(\frac{diag(|\boldsymbol{u}_0|)^{-1} + G^tG}{C}\right)\boldsymbol{u} = G^t[\boldsymbol{y} + (G\boldsymbol{u} - \boldsymbol{y} - \gamma_L \boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u} - \gamma_R \boldsymbol{e})_+] \tag{6.11}$$

This leads to the iterative method of solving the smooth HSVR problem (6.11) (SHSVR2): for i=0,1...

$$\boldsymbol{u}^{i+1} = \left(\frac{diag(|\boldsymbol{u}_0|)^{-1} + G^tG}{C}\right)^{-1} G^t[\boldsymbol{y} + (G\boldsymbol{u} - \boldsymbol{y} - \gamma_L \boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u} - \gamma_R \boldsymbol{e})_+]$$

Remark 6.4. The solution of SHSVR2 depends on the choice of the vector $\boldsymbol{u}_0$. To make $p_2(\boldsymbol{u}, \boldsymbol{u}_0) \approx \boldsymbol{u}_+$, the simple iterative procedure adopted in (Lee & Mangasarian, 2001) is followed in this work wherein the vector $\boldsymbol{u}_0$ is modified at each iteration to become closer to $|\boldsymbol{u}| = (|u_1|, \ldots, |u_{m+1}|)^t$ ((Lee & Mangasarian, 2001)).

As the final method of solving the HSVR problem (6.6), the approach on the application of a generalized derivative (Fung, 2003) for the non-smooth 'plus' function will be considered.

Taking the left-hand side derivative as a generalized derivative and hence $\frac{dx_+}{dx} = sign(x_+)$, a generalized gradient of $L(\mathbf{u})$ denoted by $\partial L(\mathbf{u})$ can be obtained as

$$\partial L(\boldsymbol{u}) = 2sign(\boldsymbol{u}_+) - \boldsymbol{e}_1 + CG^t[G\boldsymbol{u} - (\boldsymbol{y} + (G\boldsymbol{u} - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u} - \gamma_R\boldsymbol{e})_+)] \tag{6.12}$$

where $sign(\boldsymbol{u}_+)$ is a vector in $R^{(m+1)}$ whose ith component is $sign(u_{i+})$. It is easy to verify that finding the critical point of $L(\boldsymbol{u})$ using the generalized gradient (6.12) simplifies in to solving

$$G^tG\boldsymbol{u} = \frac{\boldsymbol{e}_1 - 2sign(\boldsymbol{u}_+)}{C} + G^t[\boldsymbol{y} + (G\boldsymbol{u} - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u} - \gamma_R\boldsymbol{e})_+)] \tag{6.13}$$

This suggests that the iterative method of solving HSVR by generalized derivative (GHSVR) problem (6.13) is: for i=0,1,…

$$\boldsymbol{u}^{i+1} = (G^tG)^{-1}[\frac{\boldsymbol{e}_1 - 2sign(\boldsymbol{u}_+^i)}{C} + G^t[\boldsymbol{y} + (G\boldsymbol{u}^i - \boldsymbol{y} - \gamma_L\boldsymbol{e})_+ - (\boldsymbol{y} - G\boldsymbol{u}^i - \gamma_R\boldsymbol{e})_+)]]$$

6.3 Experimental Results

In this section, we perform numerical experiments on a number of synthetic and real-world data sets and investigate the effectiveness of our proposed methods by comparing their results with SVR and LS-SVR.

All the algorithms considered were implemented in MATLAB R2008b running on a PC having 8 GB of RAM with Windows 7 OS and a 64 bit, 3.20 GHz Intel(R) Core^{TM} i5-3470 processor. For SVR, we used the MOSEK optimization tools for MATLAB available at http://www.mosek.com. The Gaussian kernel of the form: $k(\boldsymbol{x},\boldsymbol{z}) = exp(-\mu||\boldsymbol{x} - \boldsymbol{z}||^2)$ is taken, where $\mu > 0$ is the kernel parameter. For evaluating the test accuracy, the 2-norm root mean square error (RMSE) is chosen. For the termination of the training of the algorithms, i.e., SHSVR1, SHSVR2 and GHSVR, the maximum number of iterations and the minimum learning accuracy were taken as 20 and 10^{-4} respectively. The values of the regularization parameter C and the kernel parameter μ were selected from the sets $\{10^{-5}, 10^{-4}, \ldots, 10^5\}$ and $\{2^{-5}, 2^{-4}, \ldots, 2^5\}$ respectively. ε_L, ε_R were chosen from $\{0.001, 0.01, 0.1\}$ and however, following (Balasundaram & Meena, 2018) γ_L, γ_R were selected from $\{0.1, 1.0, 1.345\}$. All the parameters were obtained by employing 10-fold cross-validation procedure. Using these values, the RMSE on the test data set was calculated.

6.3.1 Synthetic Datasets

In order to test the robustness of the proposed methods, experiments were conducted on synthetic data sets generated by functions whose outputs corresponding to the training samples were polluted by adding noise drawn from either a uniform or a Gaussian distribution, and their effects in terms of test accuracy were investigated.

Let represent the uniform probability distribution in the interval and let the Gaussian distribution with mean value and standard deviation equals to be denoted by. For our experimental study, we generated five data sets using functions having 200 training and 1000 test samples and were selected randomly by uniform distribution. For each function, the observed value corresponding to the training sample was taken to be: and where is drawn uniformly from the domain of definition of and is the noise. 1000 test samples were randomly chosen, however, free of noise. In Table 6.1, we have listed the functions used for generating the data sets and the type of additive noise applied.

To avoid biased comparisons, noisy samples were randomly generated ten times and their averaged accuracies on the test data sets were listed in Table 6.2 and Table 6.3 for linear and Gaussian kernel respectively. It can be observed from the results that good generalization is achieved by the proposed methods. This suggests the suitability and usefulness of the proposed Huber SVR via 1-norm regularization.

Table 6.1 List of functions used for generating data sets and the details of additive noise

Name	Function definition	Domain of definition	Details of additive noise
Function1	$0.02(12 + 3x - 3.5x^2 + 7.2x^3)$ $(1 + cos4\pi x)(1 + 0.8sin3\pi x)$	$x \in [0,1]$	$U[-0.25, 0.25]$
Function2	$\frac{40e^{8\{(x_1-0.5)^2+(x_2-0.5)^2\}}}{e^{\{8(x_1-0.2)^2+(x_2-0.7)^2\}}+e^{\{8(x_1-0.7)^2+(x_2-0.2)^2\}}}$	$x_1, x_2 \in [0,1]$	$U[-0.2, 0.2]$
Function3	$\frac{\sin\sqrt{x_1^2+x_2^2}}{\sqrt{x_1^2+x_2^2}}$	x_1, x_2 $\in [-4\pi, 4\pi]$	$N[0, 0.1^2]$
Function4	$1.3356\left(e^{(3(x_2-0.5))}\sin(4\pi(x_2-0.9)^2)\right.$ $+ 1.5 (1 - x_1)$ $\left.+ e^{(2x_1-1)}\sin(4\pi(x_1 - 6)^2)\right)$	$x_1, x_2 \in [-0.5,0.5]$	$N[0, 5^2]$
Function5	$tan^{-1}\left[\frac{x_2x_3}{x_1} - \frac{1}{x_2x_4}\right]$	$x_1 \in [0,100]$ $x_2 \in [40\pi, 560\pi]$ $x_3 \in [0,1]$ $x_4 \in [1,11]$	$N[0, 0.1]$

Table 6.2 Comparison of the results of SHSVR1, SHSVR2 and GHSVR with SVR and LS-SVR on synthetic data sets for Linear kernel. The best result is shown in bold type.

Dataset (Train Size, Test Size)	SVR RMSE (C,ε)	LS-SVR RMSE (C)	SHSVR1 RMSE (C,γ_L,γ_R)	SHSVR2 RMSE (C,γ_L,γ_R)	GHSVR RMSE (C,γ_L,γ_R)
Function1 (200×1,1000×1)	**0.2931** $(10^{-3},10^{-5})$	0.2988 (10^{-1})	0.2982 $(10^{3}, 1,1)$	0.2981 $(10^{0}, 1,1)$	0.2982 $(10^{5}, 1,1)$
Function2	0.1192	**0.1172**	**0.1172**	0.1174	0.1174

(200×2,1000×2)	$(10^{0},10^{-1})$	(10^{0})	$(10^{3}, 1,1)$	$(10^{0}, 1,1)$	$(10^{1}, 1,1)$
Function3 (200×2,1000×2)	0.1171 $(10^{0},10^{-1})$	**0.1159** (10^{1})	0.1162 $(10^{1}, 1,1)$	0.1167 $(10^{0}, 1,1)$	0.1164 $(10^{1}, 1,1)$
Function4 (200×2,1000×2)	0.3514 $(10^{-5},10^{-1})$	0.3447 (10^{0})	**0.3446** $(10^{3}, 1,1)$	**0.3446** $(10^{5}, 1,1)$	**0.3446** $(10^{1}, 1,1)$
Function5 (200×4,1000×4)	0.0968 $(10^{-1},10^{-1})$	**0.0962** (10^{5})	**0.0962** $(10^{0}, 1,1)$	**0.0962** $(10^{5}, 1,1)$	**0.0962** $(10^{5}, 1,1)$

Table 6.3 Comparison of the results of SHSVR1, SHSVR2 and GHSVR with SVR and LS-SVR on synthetic data sets for Gaussian kernel. The best result is shown in bold type.

Dataset (Train Size, Test Size)	SVR RMSE (C,μ,ε)	LS-SVR RMSE (C,μ)	SHSVR1 RMSE $(C,\mu,\gamma_L,\gamma_R)$	SHSVR2 RMSE $(C,\mu,\gamma_L,\gamma_R)$	GHSVR RMSE $(C,\mu,\gamma_L,\gamma_R)$
Function1 (200×1, 1000×1)	0.1405 $(10^{5},2^{4},10^{-1})$	0.1391 $(10^{5},2^{4})$	0.1390 $(10^{4}, 2^{5},1,1)$	**0.1388** $(10^{5}, 2^{5},1,1)$	0.1389 $(10^{4}, 2^{5},1,1)$
Function2 (200×2, 1000×2)	0.1126 $(10^{5},2^{-2},10^{-3})$	0.1126 $(10^{5},2^{0})$	**0.1125** $(10^{4}, 2^{1},0.1,0.1)$	0.1126 $(10^{5}, 2^{5},0.1,1)$	**0.1125** $(10^{5},2^{4},0.1,0.1)$
Function3 (200×2, 1000×2)	0.1957 $(10^{2},2^{3},10^{-3})$	0.2044 $(10^{2},2^{3})$	**0.1955** $(10^{5},2^{3},0.1,0.1)$	0.2025 $(10^{2},2^{4},0.1,0.1)$	**0.1955** $(10^{5},2^{3},0.1,0.1)$
Function4 (200×2, 1000×2)	0.1107 $(10^{0},2^{5},10^{-3})$	0.1115 $(10^{1},2^{5})$	0.1093 $(10^{5},2^{2},0.1,0.1)$	0.1097 $(10^{2},2^{4},0.1,0.1)$	**0.1092** $(10^{4},2^{2},0.1,0.1)$
Function5 (200×4, 1000×4)	**0.0003** $(10^{1},2^{1},10^{-3})$	**0.0003** $(10^{4},2^{1})$	**0.0003** $(10^{5}, 2^{0},0.1,1)$	**0.0003** $(10^{5},2^{1},0.1,0.1)$	0.0004 $(10^{5},2^{5},0.1,0.1)$

6.3.2 Real World Benchmark Datasets

For further analysis, we performed experiments on eleven benchmark real-world data sets: They are Auto price, Fertility, Triazines, Boston housing data sets from UCI repository at http://www.ics.uci.edu/~mlearn; Hydraulic actuator data set (Fung & Mangasarian, 2003); Pollution, Balloon, Quake data sets from Statlib collection http://lib.stat.cmu.edu/datasets; Demo data set from http://www.toronto.edu/~delve/data and also the financial data sets IBM, Intel from http://finance.yahoo.com. From the 755 closing prices considered from 01-01-2006 to 31-12-2008 for the financial data sets and using window size equals to five, 750 samples were obtained.

For each data set, as preprocessing, normalization is performed on the value of each attribute of a sample so that it lies in the interval $(0,1)$. By randomly dividing the whole data set into 10 equal parts, and taking one of them for testing and the remaining parts for training, test accuracy in terms of RMSE is computed. Finally, the averaged RMSE and the standard deviation of the test accuracies are tabulated.

The hydraulic actuator dataset (Fung & Mangasarian, 2003) used for nonlinear system identification, consists of 1024 pair of values $(u(t), y(t))$ where $u(t)$ is the size of the valve through which oil flows into the actuator and $y(t)$ is the oil pressure. Taking $\boldsymbol{x}(t) = (y(t-1), y(t-2), y(t-3), u(t-1),\ u(t-2))^t$ (Fung, 2003) whose output is $y(t)$, one can get 1021 samples of the form: $(\boldsymbol{x}(t), y(t))$.

As examples, the accuracy and the prediction error plots for Hydraulic actuator and Auto price data set for linear and Gaussian kernals are illustrated in Figure 6.1 to Figure 6.8. Table 6.4 and Table 6.5 show the results of comparison on the performance for the proposed algorithms with SVR and LS-SVR in terms of test accuracy along with the optimal parameter values and the learning time using Linear and Gaussian kernel respectively. From the figures and table, it can be observed that the proposed algorithms show comparable generalization performance in terms of test accuracy.

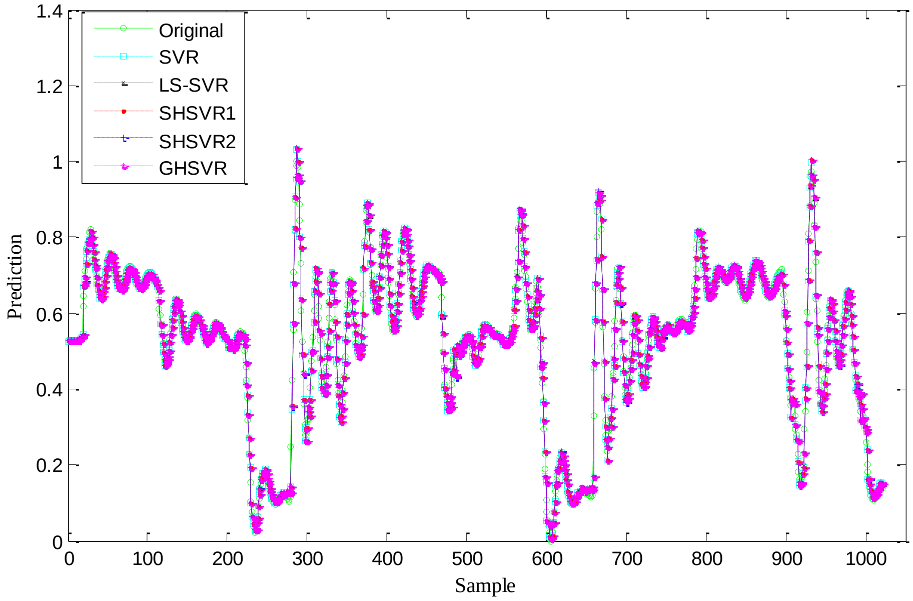


Figure 6.1 Prediction plots for Hydraulic actuator dataset for Linear kernel.

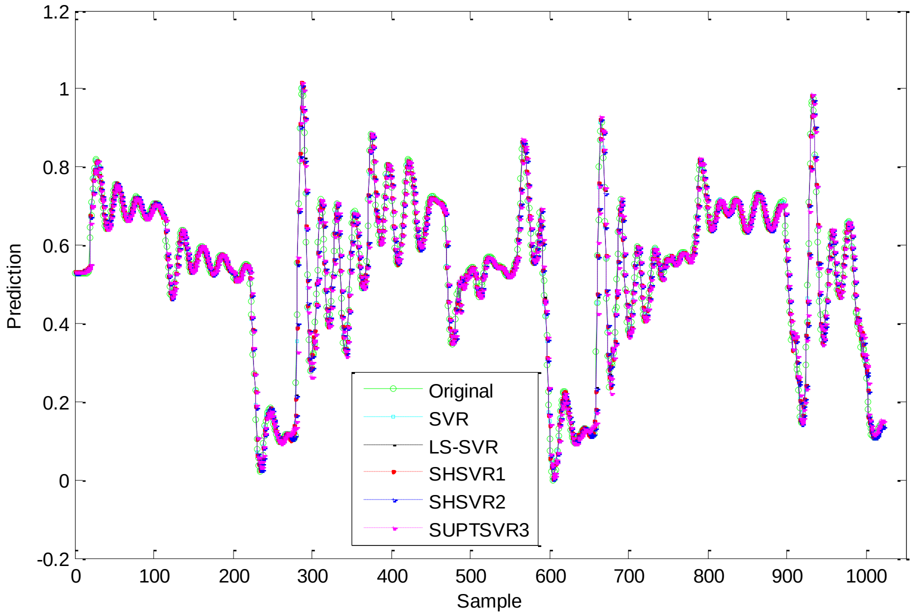


Figure 6.2 Prediction plots for Hydraulic actuator dataset for Gaussian kernel.

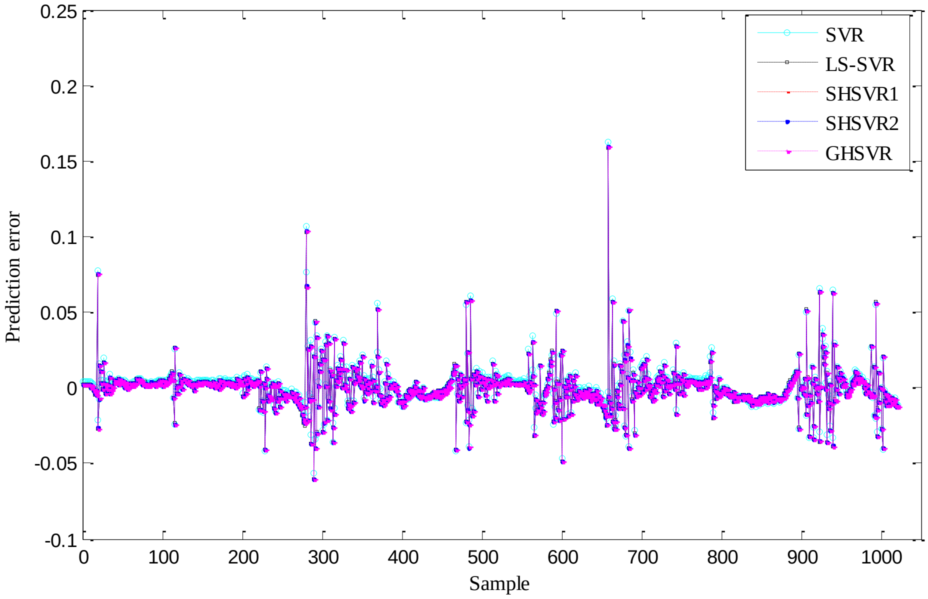


Figure 6.3 Prediction error plots for Hydraulic actuator dataset for Linear kernel.

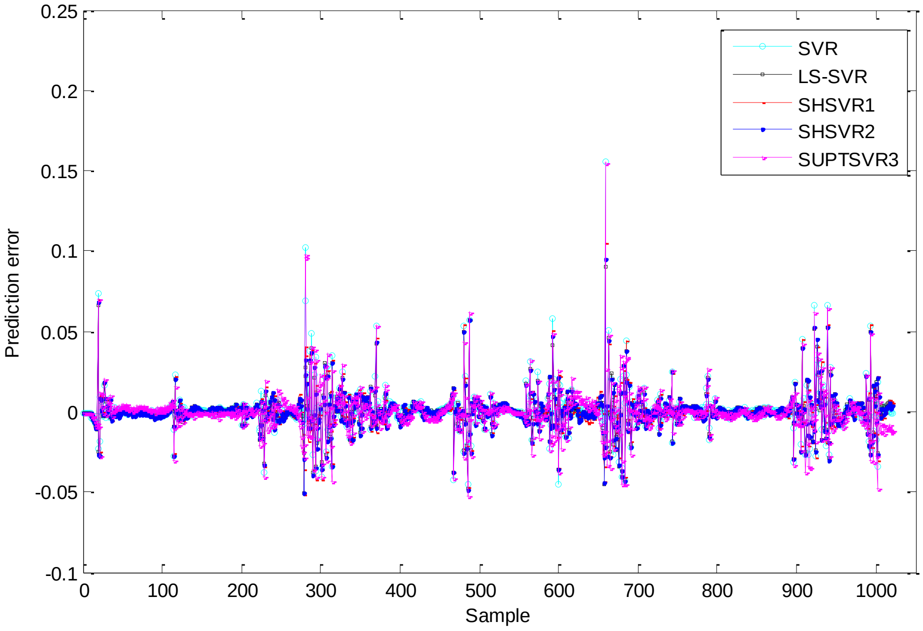


Figure 6.4 Prediction error plots for Hydraulic actuator dataset for Gaussian kernel.

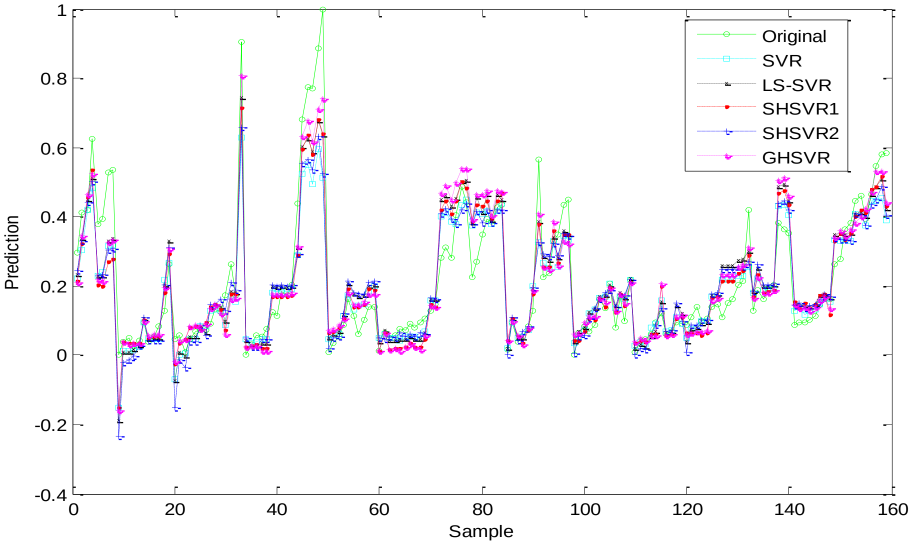


Figure 6.5 Prediction plots for auto price dataset for Linear kernel.

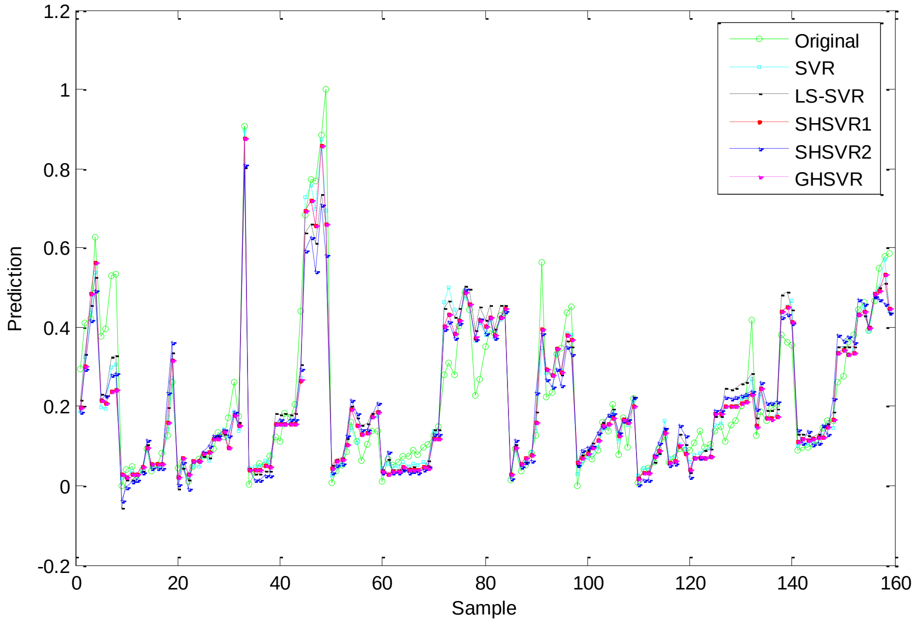


Figure 6.6 Prediction plots for auto price dataset for Gaussian kernel.

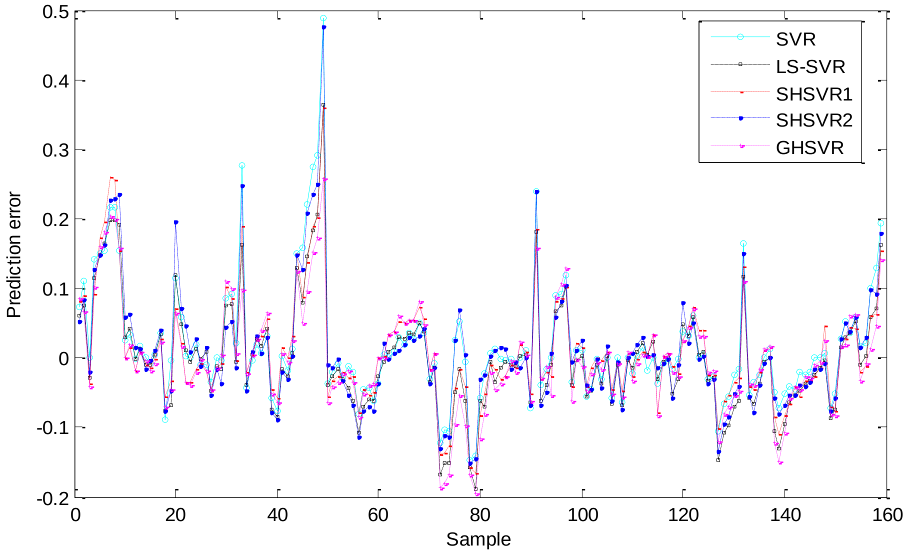


Figure 6.7 Prediction error plots for auto price dataset for Linear kernel

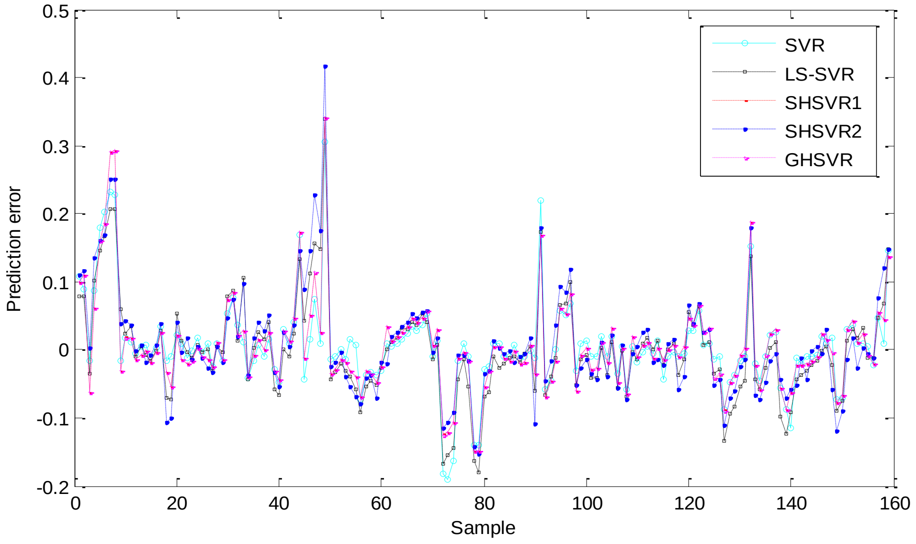


Figure 6.8 Prediction error plots for auto_price dataset for Gaussian kernel

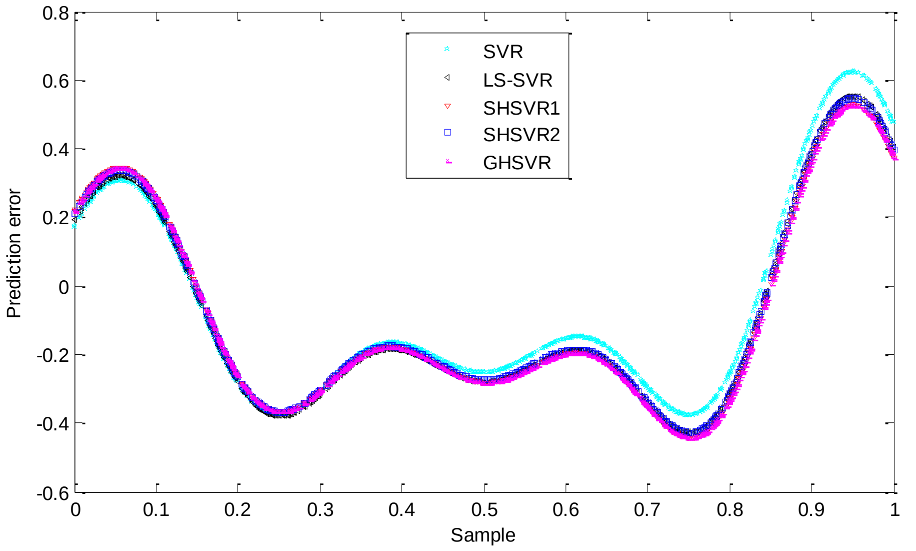


Figure 6.9 Prediction error plots for Function1 where Uniform noise was employed

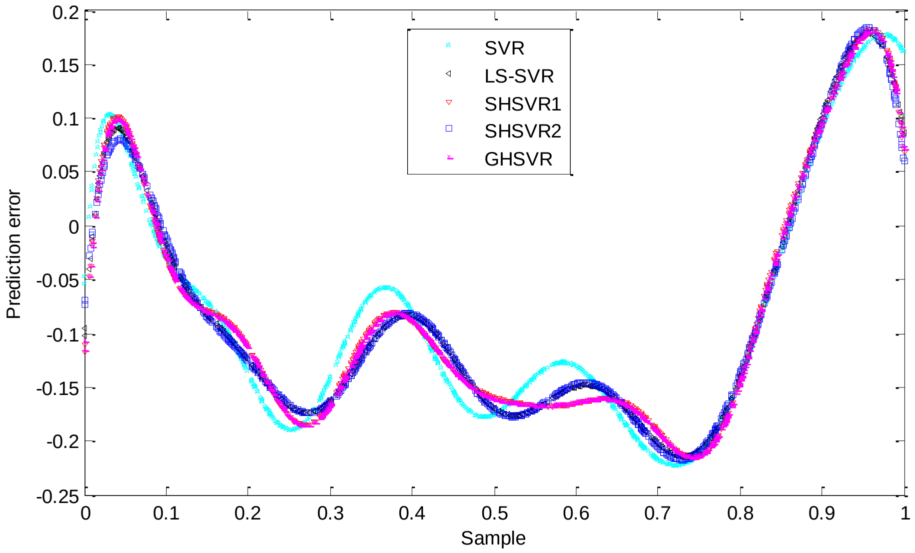


Figure 6.10 Prediction error plots for Function1 where Gaussian noise was employed

Table 6.4 Comparison of the results of SHSVR1, SHSVR2 and GHSVR with SVR and LS-SVR and the average ranks of the results on real world data sets. Test accuracy was calculated using RMSE. Linear kernel is used. The best result is shown in bold type.

Dataset (Size)	SVR RMSE (C,ε)	LS-SVR RMSE (C)	SHSVR1 RMSE (C,γ_L,γ_R)	SHSVR2 RMSE (C,γ_L,γ_R)	GHSVR RMSE (C,γ_L,γ_R)
Hydraulic actuator (1021×5)	0.0130±0.0052 $(10^1,10^{-2})$	**0.0129**±0.0054 (10^5)	**0.0129**±0.0054 $(10^5,1,0.1)$	**0.0129**±0.0054 $(10^5,0.1,0.1)$	**0.0129**±0.0054 $(10^4,0.1,0.1)$
Auto_price (159×15)	**0.0900**±0.0395 $(10^{-1},10^{-3})$	0.0935±0.0362 (10^0)	0.1013±0.0491 $(10^3,1,0.1)$	0.0951±0.0393 $(10^1,1,0.1)$	0.1015±0.0438 $(10^3,1,1)$
Fertility (99×9)	**0.2792**±0.2155 $(10^0, 10^{-3})$	0.2991±0.1446 (10^{-2})	0.2812±0.2033 $(10^5,0.1,1)$	0.2811±0.2041 $(10^1,0.1,1)$	0.2813±0.2033 $(10^5,0.1,0.1)$
Pollution (60×15)	0.1167±0.0350 $(10^0,10^{-1})$	0.1214±0.0404 (10^0)	0.1293±0.0477 $(10^5,1,0.1)$	**0.1206**±0.0331 $(10^1,1,0.1)$	0.1293±0.0477 $(10^5,1,0.1)$
Quake (2178×3)	0.1807±0.0181 $(10^5, 10^{-3})$	**0.1714**±0.0132 (10^{-1})	**0.1714**±0.0132 $(10^4,1,1)$	**0.1714**±0.0132 $(10^1,1,1)$	**0.1714**±0.0132 $(10^5,1,1)$

Balloon (2001X1)	0.0634±0.0167 $(10^{3}, 10^{-2})$	0.0534±0.0255 (10^{5})	0.0501±0.0217 $(10^{-1},1,0.1)$	0.0529±0.0259 $(10^{5},1,0.1)$	**0.0454**±0.0175 $(10^{-1},01,0.1)$
Servo (167X4)	0.1612±0.0379 $(10^{2},10^{-1})$	**0.1600**±0.0314 (10^{1})	**0.1600**±0.0315 $(10^{2},1,1)$	**0.1600**±0.0314 $(10^{2},1,1)$	0.1601±0.0314 $(10^{2},1,1)$
Demo (2018X4)	0.1024±0.0119 $(10^{3}, 10^{-3})$	**0.0999**±0.0119 $(10^{3}$, 10-3)	**0.0999**±0.0106 $(10^{3},1,1)$	**0.0999**±0.0106 $(10^{1},1,1)$	**0.0999**±0.0106 $(10^{1},1,1)$
Triazines (186X58)	**0.1715**±0.0547 $(10^{-1}, 10^{-3})$	0.1722±0.0500 (10^{-1})	0.1934±0.0407 $(10^{5},1,1)$	0.1705±0.0569 $(10^{1},0.1,0.1)$	0.1932±0.0404 $(10^{4},1,1)$
Boston housing (506X13)	0.1090±0.0651 $(10^{0},10^{-2})$	0.1061±0.0235 (10^{0})	0.1102±0.0677 $(10^{4},0.1,0.1)$	0.1091±0.0617 $(10^{1},0.1,0.1)$	**0.1051**±0.0853 $(10^{0},1,0.1)$
Sunspots (290X5)	0.0874±0.0282 $(10^{1},10^{-3})$	0.0874±0.0230 (10^{3})	**0.0868**±0.0255 $(10^{3},0.1,0.1)$	0.0869±0.0255 $(10^{5},0.1,0.1)$	0.0869±0.0255 $(10^{5},0.1,0.1)$
IBM (750X5)	**0.0240**±0.0132 $(10^{1},10^{-2})$	**0.0240**±0.0131 (10^{2})	0.0247±0.0131 $(10^{4},0.1,0.1)$	**0.0240**±0.0131 $(10^{3},0.1,0.1)$	**0.0240**±0.0131 $(10^{5},0.1,0.1)$
Intel (750X5)	**0.0285**±0.0097 $(10^{0},10^{-2})$	0.0286±0.0098 (10^{2})	0.0286±0.0098 $(10^{5},0.1,0.1)$	0.0286±0.0098 $(10^{5},0.1,0.1)$	0.0286±0.0098 $(10^{5},0.1,0.1)$
Citigroup (750X5)	**0.0138**±0.0057 $(10^{2},10^{-3})$	**0.0138**±0.0057 (10^{3})	**0.0138**±0.0057 $(10^{5},0.1,0.1)$	**0.0138**±0.0057 $(10^{5},0.1,1)$	**0.0138**±0.0057 $(10^{4},0.1,1)$

Table 6.5 Comparison of the results of SHSVR1, SHSVR2 and GHSVR with SVR and LS-SVR on real world data sets. Test accuracy was calculated using RMSE. Gaussian kernel is used. The best result is shown in bold type.

Dataset (Size)	SVR RMSE (C,μ,ε)	LS-SVR RMSE (C,μ)	SHSVR1 RMSE $(C,\mu,\gamma_L,\gamma_R)$	SHSVR2 RMSE $(C,\mu,\gamma_L,\gamma_R)$	GHSVR RMSE $(C,\mu,\gamma_L,\gamma_R)$
Hydraulicactuator (1021X5)	0.0117±0.0051 $(10^{1},2^{1},10^{-3})$	**0.0112**±0.0046 $(10^{2}, 2^{2})$	0.0113±0.0046 $(10^{5},2^{1},0.1,1)$	0.0113±0.0046 $(10^{4},2^{2},0.1,0.1)$	**0.0112**±0.0047 $(10^{5},2^{1},0.1,1)$

Auto_price (159✗15)	0.0850±0.0394 $(10^1,2^{-4},10^{-2})$	0.0897±0.0366 $(10^1, 2^{-4})$	0.0920±0.0413 $(10^5,2^{-5},1,0.1)$	**0.0840**±0.0338 $(10^2,2^{-5},1,0.1)$	0.0920±0.0413 $(10^5,2^{-5},1,0.1)$
Fertility (99✗9)	0.2805±0.2111 $(10^0,2^{-5},10^{-2})$	0.2938±0.1408 $(10^{-1}, 2^0)$	0.2797±0.1944 $(10^4,2^{-5},0.1,1)$	**0.2654**±0.1973 $(10^1,2^2,0.1,0.1)$	0.2800±0.1941 $(10^4,2^{-5},0.1,1)$
Pollution (60✗15)	**0.1108**±0.0352 $(10^0,2^{-3},10^{-2})$	0.1155±0.0395 $(10^1, 2^{-2})$	0.1194±0.0355 $(10^4,2^{-5},1,0.1)$	0.1074±0.0329 $(10^2,2^{-3},1,0.1)$	0.1211±0.0349 $(10^4,2^{-5},1,0.1)$
Quake (2178✗3)	0.1759±0.0143 $(10^1,2^1,10^{-1})$	**0.1714**±0.0132 $(10^{-1}, 2^0)$	0.1715±0.0131 $(10^5,2^{-3},1,1)$	**0.1714**±0.0124 $(10^0,2^5,1,1)$	0.1715±0.0131 $(10^5,2^{-3},1,1)$
Balloon (2001✗1)	**0.0430**±0.0273 $(10^0,2^0,10^{-1})$	0.0459±0.0235 $(10^1, 2^0)$	0.0458±0.0233 $(10^4,2^{-2},1,1)$	0.0453±0.0240 $(10^1,2^1,1,0.1)$	0.0459±0.0232 $(10^4,2^{-2},1,1)$
Servo (167✗4)	**0.0779**±0.0506 $(10^2,2^{-1},10^{-3})$	0.0815±0.0238 $(10^3, 2^1)$	0.0810±0.0422 $(10^5,2^{-1},1,0.1)$	0.0811±0.0422 $(10^5,2^{-1},1,0.1)$	0.0811±0.0422 $(10^5,2^{-1},1,0.1)$
Demo (2018✗4)	0.0886±0.0097 $(10^0,2^4,10^{-2})$	**0.0854**±0.0093 $(10^0, 2^4)$	0.0885±0.0088 $(10^4,2^0,1,1)$	0.0861±0.0093 $(10^1,2^5,1,1)$	0.0885±0.0088 $(10^4,2^0,1,1)$
Triazines (186✗58)	0.1736±0.0516 $(10^2,2^{-5},10^{-3})$	**0.1635**±0.0561 $(10^1, 2^{-3})$	0.1672±0.0548 $(10^5,2^{-5},0.1,1)$	0.1644±0.0583 $(10^3,2^{-4},0.1,1)$	0.1672±0.0548 $(10^5,2^{-5},0.1,1)$
Boston housing (506 X 13)	0.0874±0.0567 $(10^2,2^{-5},10^{-2})$	0.0877±0.0532 $(10^2, 2^{-4})$	0.0855±0.0580 $(10^5,2^{-5},1,0.1)$	**0.0833**±0.0509 $(10^3,2^{-3},0.1,0.1)$	0.0854±0.0581 $(10^5,2^{-5},1,0.1)$
Sunspots (290✗5)	0.0722±0.0147 $(10^1,2^{-1},10^{-2})$	**0.0708**±0.0137 $(10^1, 2^2)$	0.0721±0.0123 $(10^4,2^{-1},0.1,0.1)$	0.0711±0.0109 $(10^3,2^1,1,1)$	0.0721±0.0131 $(10^4,2^{-1},1,0.1)$
IBM (750✗5)	**0.0241**±0.0132 $(10^2,2^{-5},10^{-2})$	0.0242±0.0129 $(10^4, 2^{-5})$	0.0242±0.0130 $(10^5,2^{-3},0.1,0.1)$	**0.0241**±0.0130 $(10^5,2^{-3},0.1,0.1)$	**0.0241**±0.0130 $(10^5,2^{-3},0.1,0.1)$
Intel (750✗5)	**0.0285**±0.0095 $(10^2,2^{-5},10^{-3})$	0.0287±0.0098 $(10^3, 2^{-5})$	**0.0285**±0.0090 $(10^5,2^{-1},0.1,0.1)$	**0.0285**±0.0090 $(10^5,2^{-1},0.1,1)$	**0.0285**±0.0092 $(10^5,2^{-2},0.1,0.1)$
Citigroup	0.0139±0.0059	0.0139±0.0059	**0.0138**±0.0060	0.0139±0.0062	**0.0138**±0.0058

(750×5)	$(10^{1},2^{-2},10^{-3})$	$(10^{5}, 2^{-4})$	$(10^{5},2^{-3},0.1,0.1)$	$(10^{5},2^{-2},0.1,1)$	$(10^{5},2^{-3},0.1,0.1)$

Table 6.6 Average ranks of SVR, LS-SVR, SHSVR1and SHSVR2 with linear kernel.

Dataset	SVR	LS-SVR	SHSVR1	SHSVR2	GHSVR
Hydraulic actuator	5	2.5	2.5	2.5	2.5
Auto_price	1	2	4	3	5
Fertility	1	5	3	2	4
Pollution	1	3	4.5	2	4.5
Quake	5	2.5	2.5	2.5	2.5
Balloon	5	4	2	3	1
Servo	5	2	2	2	4
Demo	5	2.5	2.5	2.5	2.5
Triazines	2	3	5	1	4
Boston housing	3	2	5	4	1
Sunspots	4.5	4.5	1	2.5	2.5
IBM	2.5	5	2.5	2.5	2.5
Intel	5	2.5	2.5	2.5	2.5
Citigroup	3	3	3	3	3
Average rank	3.4285	3.1071	3	2.5	2.9642

Table 6.7 Average ranks of SVR, LS-SVR, SHSVR1and SHSVR2 with Gaussian kernel.

Dataset	SVR	LS-SVR	SHSVR2	SHSVR2	GHSVR
Hydraulic actuator	5	1.5	3.5	3.5	1.5
Auto_price	2	3	4	1	4
Fertility	4	5	2	1	3

Pollution	2	3	4	1	5
Quake	5	1.5	3.5	1.5	3.5
Balloon	1	4.5	3	2	4.5
Servo	1	5	2	3.5	3.5
Demo	5	1	2.5	4	2.5
Triazines	5	1	3.5	2	3.5
Boston housing	4	5	3	1	2
Sunspots	5	1	3.5	2	3.5
IBM	2	4.5	4.5	2	2
Intel	2.5	5	2.5	2.5	2.5
Citigroup	4	4	1.5	4	1.5
Average rank	3.3928	3.2142	3.0714	2.2142	3.0357

To analyze statistically the performance of the five algorithms considered, we apply the Friedman test and, if necessary, its corresponding post hoc tests. For this purpose, we ranked the algorithms for each data set separately and reported their average ranks, in Table 6.7. Among the average rank, the minimum is shown by SHVR2 clearly indicates its superiority. Under the null-hypothesis that all the algorithms are equivalent, we compute the Friedman statistic on 11 real-world data sets of Table 6.4

$$\chi_F^2 = \frac{12 \times 11}{5 \times 6}\left[3.4091^2 + 3.1818^2 + 3.4091^2 + 1.7727^2 + 3.2273^2 - \frac{5 \times 6^2}{4}\right] \cong 8.4740$$

$$F_F = \frac{10 \times 8.4740}{11 \times 4 - 8.4740} \cong 2.3853$$

Here F_F is the F-distribution with $(4, 4 \times 10) = (4, 40)$ degrees of freedom. From statistical table, the critical value of $F(4,40)$ for the level of significance $\alpha = 0.05$ is 2.6060. Since the computed F_F value, i.e. 2.3853, is smaller than 2.6060 and hence we conclude that there is no significant difference between the five algorithms.

6.4 Conclusions

Although the error of misfit measured using the epsilon insensitive function of Vapnik (Vapnik, 2000) leads to robust regression model, this function is only continuous and therefore the application of popular numerical minimization methods of solving is difficult. In this chapter, Huber function is used as the error function to measure the data misfit having both the robustness and differentiability properties. Our proposed formulation leads to solving an optimization problem whose solution was obtained by functional iterative methods. Tests with both synthetic and real-world data sets confirm the suitability and applicability of our proposed robust model.

Chapter 7

Contribution and Future Work

7.1 The Thesis's Contribution

In this thesis, we have studied the design and analysis of algorithms for nonparallel SVM with pinball loss function for pattern classification and Huber loss function based SVM for data regression problems, with the goal of achieving comparable or better generalization performance for noise corrupted datasets while improving computational efficiency over other state-of-the-art SVM solvers.

In chapter 3, we presented a robust twin bounded support vector machine with pinball loss (Pin-TBSVM) for datasets polluted by feature noise. Pin-TBSVM is a non-parallel classifier where kernel generated surfaces were determined as the solutions of quadratic programming problems. Experimental results on several benchmark datasets show that the proposed method achieves improved accuracy performance than the popular traditional methods. Though Pin-TBSVM is a simple and efficient learning method, it loses sparsity. An increase in the number of parameters is a concern and the selection of their optimal values is a practical problem that needs attention. The proposed method's application to multi-category classification problems is interesting and worth investigating.

The next work presents a L_1-norm based twin bounded support vector machine with pinball loss for data categorization with the goal of obtaining an efficient robust learning model. Besides the benefit of less sensitivity to noise property of pinball loss, with the application of L_1-norm for within-class scatter minimization, the proposed method also enjoys robustness to outliers. As a novel approach of solving, by a simple reformulation of the primal problem considered, an equivalent pair of dual QPPs in *m* variables only is derived (L1-Pin-TBSVM), where *m* is the number of training vectors. In comparison with the twin bounded support vector machine (TBSVM), the duals of our proposed L1-Pin-TBSVM are free of inverse matrices and the non-linear duals can be obtained from their linear formulations directly by applying the kernel trick. Experiments on *Crossplanes* dataset where two or four outliers were introduced show that the proposed L1-Pin-TBSVM outperforms the other SVM methods in terms of accuracy, confirming its robustness to outliers. In addition, experimental results on a two-moon synthetic dataset and several benchmark datasets with different levels of noise clearly demonstrate improved generalization ability of L1-Pin-TBSVM at comparable training cost which further confirms its effectiveness and suitability where robustness is a problem of major concern. Though L1-Pin-TBSVM is a simple, efficient learning method, it loses sparsity. An increase in the number of parameters in our L1-Pin-TBSVM is also a concern as the selection of their optimal values is a practical problem that needs attention. As a future research, application of optimization methods for large scale datasets like sequential minimal optimization (SMO), dual coordinate descent (DCD) and successive over-relaxation (SOR) is an important practical problem worthy of consideration. Extension to semi-supervised learning is interesting and will be investigated in our future work.

In chapter 5, with the aim of having the integrated merits of L_1-norm and pinball loss in achieving enhanced robustness to outliers and bringing noise insensitivity to feature noise, we presented a novel, efficient L_1-norm based non-parallel support vector machine classifier with pinball loss (L1-Pin-NPSVM) where its associated optimization problem minimizes the scatter loss and the misclassification error by L_1-norm and pinball loss respectively. The dual formulation of the proposed method solves a pair of QPPs free of inverse kernel matrices. Our formulation allows a unified framework for the linear and nonlinear kernels. The effectiveness of L1-Pin-NPSVM is evaluated on synthetic and UCI benchmark datasets having outliers and/or being contaminated by noise. The results confirm its superiority in terms of robustness to outliers and noise insensitivity. In summary, we can conclude that the proposed L1-Pin-NPSVM is an efficient method than the other machine-learning methods for real-world problems when robustness and noise insensitivity is of major concern. Even though L1-Pin-NPSVM is a simple, efficient learning method, it loses sparsity. As a future work, an efficient method of solving L1-Pin-NPSVM for problems with large samples will be explored. The possibility of the proposed method’s application for multi-category classification will be interesting and it will be investigated as another future work.

In chapter 6, we proposed a novel robust Huber SVR (HSVR) formulation in primal where the regressor is made as flat as possible by introducing the regularization term in L1-norm. Since the regularization term is non-smooth, it is proposed to replace it with smooth approximation functions and solve the problems by functional iterative method. Tests with both synthetic and real-world datasets confirm the suitability and effectiveness of the proposed robust model.

7.2 Extensions and Future Work

We identify the following apparent extensions and future study directions as a result of the work presented in this thesis.

- This thesis created robust regression models based on the Huber loss function. Nonconvex loss functions have received greater attention in recent years due to their relative advantage over convex loss functions in terms of generalization performance and robustness (Colobrot et al., 2006). Extending these models with a truncated Huber loss function will be interesting (Zhao & Sun, 2010).

- The proposed Pin-TBSVM method's application to multi-category classification problems is interesting and worth investigating.

- As future research of L1-Pin-TBSVM, application of optimization methods for large scale datasets like sequential minimal optimization (SMO), dual coordinate descent (DCD) and successive over-relaxation (SOR) is an important practical problem worthy of consideration. Extension to semi-supervised learning is interesting and will be investigated in our future work.

- In future work, an efficient method of solving L1-Pin-NPSVM for problems with large samples will be explored. The possibility of the extension of the proposed method for multi-category classification will be interesting and it will be investigated as another future work.

- Learning algorithms for large data is a difficult problem to solve. Future studies could include extending the presented solutions to challenges with extremely big datasets.

References

Abe S., (2005). *Support Vector Machines for Pattern Classification*. Springer.

Alpaydin, E., (2014). *Introduction to Machine Learning*, 3rd ed., MIT Press.

Balasundaram S., Meena, Y., (2018). On robust regularized support vector regression in primal with asymmetric Huber loss, *Neural Processing Letters*, DOI: 10.1007/s11063-018-9875-8.

Balasundaram S., Meena, Y., (2019). Robust regularized support vector regression in primal with asymmetric Huber loss, *Neural Processing Letters*, 49, 1–33, DOI: 10.1007/s11063-018-9875-8.

Balasundaram, S., Gupta, D., Prasad, S.C., (2017). A new approach for training Lagrangian twin support vector machine via unconstrained convex minimization, *Applied Intelligence*, 46: 124-134.

Balasundaram, S., Kapil, (2010). On Lagrangian support vector regression. *Expert Systems with Applications*, 37(12):8784-8792.

Balasundaram, S., Kapil, (2011). Finite Newton method for implicit Lagrangian support vector regression. *International Journal of Knowledge-based and Intelligent Engineering System*s, 15(4):203-214.

Balasundaram, S., Singh, R., (2010). On finite Newton method for support vector regression, *Neural Computing & Applications*, 19(7):967-977.

Balasundaram, S., Tanveer, M., (2013a). On Lagrangian twin support vector regression, *Neural Computing & Applications*, 22(1):257-267.

Balasundaram, S., Tanveer, M., (2013b). Smooth Newton method for implicit Lagrangian twin support vector regression, *International Journal of Knowledge-based and Intelligent Engineering Systems*, 17(4):267-278.

Basu, A., Walters, C., Shepherd, M., (2003). Support vector machines for text categorization, *in: Proceedings of the 36th Annual Hawaii International Conference on Systems Sciences*, IEEE.

Beaumont, C. N., Williams, J. P., & Goodman, A. A. (2011). Classifying Structures in the Interstellar Medium with Support Vector Machines: The G16. 05-0.57 Supernova Remnant. *The Astrophysical Journal*, 741(1), 14.

Ben-David, S., & Lindenbaum, M., (1997). Learning distributions by their density levels: A paradigm for learning without a teacher. *Journal of Computer and System Sciences*, *55*(1):171-182.

Boser, B.E., Guyon, I.M., Vapnik, V.N., (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*. 144-152.

Burges, C.J., Schölkopf, B., (1997). Improving the accuracy and speed of support vector machines. In *Advances in neural information processing systems*. 375-381.

Burges, C.J.C., (1998). Geometry and invariance in kernel based methods, In *Advances in Kernel Methods-Support Vector Learning*, Bernhard Scholkopf, Cristopher J.C. Burges and Alexander J. Smola (eds.), MIT Press, Cambridge, MA.

Camps-Valls, G., Bruzzone, L., Rojo-Alvarez, J.L., (2006). Robust support vector regression for biophysical variable estimation from remotely sensed images, *IEEE Geoscience and Remote Sensing Letters*, 3(3), 339–343, DOI: 10.1109/LGRS.2006.871748.

Cawley, G.C., Talbot, N.L.C., (2002). Improved sparse least-squares support vector machines, *Neurocomputing*, 48:1025-1031.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3), 1-27.

Chen C., Li Y., Yan C., Liu G., (2017a). Least absolute deviation-based robust support vector regression, *Knowledge-Based Systems*, 131:183-194.

Chen, X., Yang, J., Liang, J., Ye, Q., (2012). Smooth twin support vector regression, *Neural Computing & Applications*, 21: 505-513.

Chen, X., Yang, J., Ye, Q., Liang, J., (2011). Recursive projection twin support vector machine via within-class variance minimization. *Pattern Recognition*, 44(10-11):2643-2655.

Chu W., Keerthi S.S., Ong C.J., (2004). Baysian support vector regression using a unified loss function, *IEEE Transactions on Neural Networks*, 15(1):29-44.

Chuang C.C., Su S.F., Jeng J.T., Hsiao C.C., (2002). Robust support vector regression networks for function approximation with outliers, *IEEE Transactions on Neural Networks*, 13(6):1322-1330.

Collobert, R., Bengio, S., (2001). SVM Torch: Support Vector Machines for Large Scale Regression Problems, *Journal of Machine Learning Research*, 1:143-160.

Cortes, C., Vapnik, V., (1995). Support-vector networks. *Machine learning*, 20(3):273-297.

Cristianini, N., & Shawe-Taylor, J., (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, Cambridge.

Demiriz, A., Bennett, K.P., Breneman, C.M., Embrechts, M.J., (2001). Support vector machine regression in chemometrics. In *Computing Science and Statistics: Proc. of the 33rd Symposium on the Interface*.

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

Dibike, Y.B., Velickov, S., Solomatine, D., (2000, March). Support vector machines: Review and applications in civil engineering. In *Proceedings.of the joint workshop on Applications of AI in Civil Engineering, Cottbus-2000, Germany.*

Ding, H., Xu, J., (2015). Random gradient descent tree: a combinatorial approach for SVM with outliers, *in: Proceedings of AAAI conference on Artificial Intelligence (AAAI)*, 2561-2567.

Fan, R.E., Chen, P.H., Lin, C.J., (2005). Working set selection using the second order information for training support vector machines, *Journal of Machine Learning Research*, 6:1889–1918.

Frenay, B., Verleysen, M., (2014). Classification in the presence of label noise: a survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5) : 845-869.

Fung, G., Mangasarian, O.L., (2002). Incremental support vector machine classification. In *Proceedings of the 2002 SIAM International Conference on Data Mining* (pp.247-260). Society for Industrial and Applied Mathematics.

Fung, G., Mangasarian, O.L., (2003). Finite Newton method for Lagrangian support vector machine, *Neurocomputing*, 55(1-2), 39-55.

Gammerman, A., Kalnishkan, Y., & Vovk, V., (2004). On-line prediction with kernels and the Complexity Approximation Principle, In Max Chickering and Joseph Halpern, editors, *Proceedings of the Twentieth Annual Conference on Uncertainty in Artificial Intelligence*, AUAI Press, Arlington, VA, pp.170–176.

Gao, S., Ye, Q., Ye, N., (2011). 1-norm least squares twin support vector machines. *Neurocomputing,* 74: 3590-3597.

Gretton A., Doucet A., Herbrich R., Rayner P.J.W., Scholkopf B., (2001). Support vector regression for black-box system identification. In: Proceedings of the 11th IEEE Workshop on Statistical Signal Processing, 341–344. IEEE, Singapore.

Gu, Z., Zhang, Z., Sun, J., Li, B., (2017). Robust image recognition by L1-norm twin-projection support vector machine, *Neurocomputing*, 223 : 1-11.

Guitton A., Symes W.W., (2003). Robust inversion of seismic data using the Huber norm, Geophysics, 68(4), 1310-1319.

Guyon I., Weston J., Barnhill S., Vapnik V.N., (2002). Gene selection for cancer classification using support vector machine. *Machine Learning*, 46: 389-422.

Hao, P.Y., (2010). New support vector algorithms with parametric insensitive/margin model. *Neural Networks*, 23(1): 60-73.

Hao, P.Y., Tsai, L.B., & Lin, M.S., (2008). A new Support vector classification algorithm with parametric-margin model, *IEEE International joint conference, IJCNN, Neural Networks*, 420-425.

Hoerl, A.E., (1962). Application of ridge analysis to regression problems, *Chemical Engineering Process*, 58, pp.54-59.

Hong, D.H., Hwang, C., & Ahn, C., (2004). Ridge Estimation for Regression Models with Crisp Inputs and Gaussian Fuzzy Output, *Fuzzy Sets and Systems*, 142, pp.307-319.

Hsieh, C-J., Chang, K-W., Lin, C-J., (2008). A dual coordinate descent method for large scale linear SVM, *in: Proceedings of the 25th International Conference on Machine Learning*, Helsinki.

Huang X., Shi L., Pelckmans K., Suykens J.A.K., (2014a). Asymmetric *v*-tube support vector regression, *Computational Statistics and Data Analysis*, 77:371-382.

Huang, X., Shi, L., Suykens, J.A.K., (2014b). Support vector machine classifier with pinball loss. *IEEE transactions on pattern analysis and machine intelligence*, *36*(5):984-997.

Huang, X., Shi, L., Suykens, J.A.K., (2014c). Asymmetric least squares support vector machine classifiers. *Computational Statistics and Data Analysis*, *70*:395-405.

Huang, X., Shi, L., Suykens, J.A.K., (2014d). Ramp loss linear programming support vector machine. *Journal of Machine Learning Research*, 15: 2185-2211.

Huber P.J., Ronchetti E.M., (2009). Robust Statistics. 2nd ed., New York, Wiley

Hull, J.J., Taylor, S.L., (1998). *Document analysis systems II* (Vol. 29). World Scientific.

Jayadeva, Khemchandani, R., Chandra, S., (2004). Fast and Robust Learning through Fuzzy Linear Proximal Support Vector Machines, *Neurocomputing*, 61:401-411.

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.

Ji, Y., & Sun, S., (2013). Multitask multiclass support vector machines: Model and experiments, *Pattern Recognition*, 46(3):914-924.

Joachims, T., (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*. Springer, Berlin, Heidelberg, 137-142.

Joachims, T., (1999). Svmlight: Support vector machine. *SVM-Light Support Vector Machine http://svmlight. joachims. org/, University of Dortmund*, *19*(4).

Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K., (2001). Improvements to Platt's SMO algorithm for SVM classifier design. *Neural computation*, *13*(3):637-649.

Kim S.K., Park Y.J., Toh K.A., Lee S., (2010). SVM-based feature extraction for face recognition. *Pattern Recognition*, 43(8): 2871–2881.

Kim, K.J., (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, *55*(1-2):307-319

Kim, S.K., Park, Y.J., Toh. K.A., Lee, S., (2010). SVM-based feature extraction for face recognition. *Pattern Recognition*, 43(8): 2871–2881.

Kodratoff Y., Michalski R.S., (2014). Machine Learning: An artificial intelligence approach, Vol. 3, Elsevier.

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

Lee, Y.J., Hsieh, W.F., Huang, C.M., (2005). ε-SSVR: A smooth support vector machine for ε-insensitive regression. *IEEE Trans. on Knowledge and Data Engineering*, 17(5):.678-685.

Lee, Y.J., Mangasarian, O.L., (2001). SSVM: A smooth support vector machine for classification. Computational Optimization and Applications, 20(1), 5–22 .

Ma, S., Cheng, B., Shang, Z., Liu, G., (2018). Scattering transform and LSPTSVM based fault diagnosis of rotating machinery, Mech.Syst. Signal Processing, 104: 155-170.

Madsen K., Nielsen H.B., (1990) Finite algorithms for robust linear regression. *BIT Numerical Mathematics* 30(4):682–699.

Mangasarian, O.L., (2002). A finite Newton method for classification, *Optimization Methods and Software*, 17:913-929.

Mangasarian, O.L., Musicant, D.R., (1999). Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, *10*(5):1032-1037.

Mangasarian O.L., Musicant D., (2000) Robust linear and support vector regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):950–955.

Mangasarian, O.L., Musicant, D.R., (2001a). Active set support vector machine classification. In T.K. Leen, T.G. Dietterich V.Tesp (Eds.), *Advances in Neural Inf. Processing Systems 13*, MIT Press,577-586.

Mangasarian, O.L., & Musicant, D.R., (2001b). Lagrangian support vector machines, *Journal of Machine Learning Research*, 1:161-177.

Mangasarian, O.L., Wild, E.W., (2006). Multisurface proximal support vector classification via generalized eigenvalues, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):69-74.

Mehrkanoon S., Huang X., Suykens J.A.K., (2014). Non-parallel support vector classifiers with different loss functions. *Neurocomputing*, 143: 294-301.

Mukherjee, S., Osuna, E., Girosi, F., (1997). Nonlinear prediction of chaotic time series using support vector machines. In *Neural Networks for Signal Processing VII. Proceedings of the 1997 IEEE Signal Processing Society Workshop* (511-520). IEEE.

Müller, K.R., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., Vapnik, V., (1999). Using support vector machines for time series prediction. *Advances in kernel methods-support vector learning*, 243-254.

Murphy, P.M., Aha, D.W., (1992). UCI repository of machine learning databases. University of California, Irvine, http://www.ics.uci.edu/~mlearn.

Musicant, D.R., & Feinberg, A., (2004). Active set support vector regression, *IEEE Transaction on Neural Networks*, 15(2):268-275.

Osuna, E., Freund, R., Girosi, F., (1997). Training support vector machines: an application to face detection. In: *IEEE conference on computer vision and pattern recognition*, 130–136.

Osuna, F., Freund, R., Girosi, F., (1997) An improved training algorithm for support vector machines, *in: Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, Amelia Island, FL, 276-285.

Pang, Y., Zhang, K., Yuan, Y., & Wang, K. (2014). Distributed object detection with linear SVMs. *IEEE transactions on cybernetics*, *44*(11), 2122-2133.

Papageorgiou, C. P., Oren, M., Poggio, T., (1998, January). A general framework for object detection. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, 555-562.

Peng, X., (2010a)., TSVR: An efficient twin support vector machine for regression. *Neural Networks*, 23(3):365-372.

Peng, X., (2010b). Primal twin support vector regression and its sparse approximation. *Neurocomputing*, 73:2846-2858.

Peng, X., (2010c). A v-twin support vector machine (v -TSVM) classifier and its geometric algorithms. *Information Sciences*, 180(20): 3863-3875.

Peng, X., (2011). TPMSVM: A novel twin parametric-margin support vector machine for pattern recognition, *Pattern Recognition*, 44:2678-2692.

Peng, X., (2012). Efficient twin parametric insensitive support vector regression model, *Neurocomputing*, 79(1):26–38.

Peng, X., Xu, D., Shen, J., (2014). A twin projection support vector machine for data regression. *Neurocomputing*, 138, 131–141.

Peng, X., Chen, D., Kong, L., (2014). A clipping dual coordinate descent algorithm for solving support vector machines. *Knowledge-Based Systems*, 71: 266-278.

Peng, X., Xu, D., Kong, L., Chen, D., (2016). $L_1 -$norm based twin support vector machine for data recognition. *Information Sciences*, 340: 86-103.

Platt, J. (1998)., Sequential minimal optimization: A fast algorithm for training support vector machines.

Platt, J., (1999). Fast training of support vector machines using sequential minimal optimization, Advances in Kernel Methods-Support Vector Learning, MIT Press, Cambridge. MA, 185-208.

Prasad, S.C., Balasundaram, S., (2021). On Lagrangian L2-norm pinball twin bounded support vector machine via unconstrained convex minimization. *Information Sciences*, 571: 279-302.

Rastogi R., Pal A., Chandra S., (2018). Generalized Pinball loss SVMs. *Neurocomputing*, 322: 151-165.

Ripley, B.D., (1996). Pattern Recognition and Neural Networks, Cambridge University Press, Cambridge.

Rychetsky, M., Ortmann, S., Glesner, M., (1999). Support vector approaches for engine knock detection. In, *International Joint Conference on Neural Networks. Proc.* (*Cat. No. 99CH36339*), 2: 969-974, IEEE.

Saunders, C., Gammerman, A., & Vovk, V., (1998). Ridge Regression Learning Algorithm in Dual Variables, *In Proceedings of the 15th International Conference on Machine Learning, ICML'98*, pp.515-521.

Schmidt, M., & Herbert, G., (1996). Speaker identification via support vector classifiers, *IEEE International Conference Proceedings on Acoustics, Speech and Signal Processing*, 1, 105-108.

Scholkopf, B., Smola, A.J., (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.

Scholkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L., (2000). New Support Vector Algorithms, *Neural Computation*, 12 (5):1207–1245.

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.

Shao, Y., Deng, N., Yang, Z., (2012). Least squares recursive projection twin support vector machine for classification, *Pattern Recognition*, 45(6):2299-2307.

Shao, Y., Wang, Z., Chen, W., Deng, N., (2013a). A regularization for the projection twin support vector machine, *Knowledge Based Systems*, 37:203-210.

Shao, Y., Wang, Z., Chen, W., Deng, N., (2013b). Least squares twin parametric-margin support vector machine for classification, *Applied Intelligence*, DOI 10.1007/s10489-013-0423 y.

Shao, Y., Zhang, C., Yang, Z., Jing, L., Deng, N., (2013c). An *ε*-twin support vector machine for regression, *Neural Computing & Applications*, 23(1):175–185.

Shao, Y., Zhang, C., Wang, X., Deng, N., (2011). Improvements on twin support vector machines, *IEEE Transaction on Neural Network*, 22(6):962-968.

Shawe-Taylor, J., & Sun, S., (2011). A review of optimization methodologies in support vector machines. *Neurocomputing* 74(17), 3609-3618.

Shen, X., Niu, L., Qi, Z., Tian, Y., (2017). Support vector machine classifier with truncated pinball loss, *Pattern Recognition*, 68 : 199–210.

Smola A.J., (1998) Regression estimation with support vector learning machines, Master's thesis. Technical Univ, Munchen, Munich, Germany.

Steinwart, I., Christmann, A., (2008). *Support vector machines*. Springer Science & Business Media.

Stoneking, D., (1999). Improving the manufacturability of electronic designs. *IEEE Spectrum*, *36*(6):70-76.

Sun, S., (2011). Multi-view Laplacian support vector machines, *Advanced Data Mining and Applications, LNCS*, 7121, 209-222.

Suykens J.A.K., De Brabanter J., Lukas L., Vandewalle J., (2002). Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 48(1): 85-105.

Suykens J.A.K., Vandewalle, J., (1999). Least squares support vector machine classifiers, *Neural Processing Letters*, 9(3):293-300.

Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific, Singapore (2002).

Tarassenko, L., Hayton, P., Cerneaz, N., Brady, M., (1995). Novelty detection for the identification of masses in mammograms.

Tay, F. E., Cao, L., (2001). Application of support vector machines in financial time series forecasting. *omega*, *29*(4):309-317.

Tian, Y., Ju, X., Qi, Z., Shi, Y., (2014). Improved twin support vector machine. *Science China Mathematics*, *57*(2):417-432.

Tian, Y., Qi, Z., Ju, X., Shi, Y., Liu, X., (2013). Nonparallel support vector machines for pattern classification. *IEEE transactions on cybernetics*, 44(7): 1067-1079.

Vapnik, V., Golowich, S.E., Smola, A.J., (1997). Support vector method for function approximation, regression estimation and signal processing. In *Advances in neural information processing systems*, 281-287.

Vapnik, V.N., (2000). *The nature of statistical learning theory*. 2nd ed., Springer, New York.

Wang, Z., Shao, Y.H., Wu, T.R., (2014). Proximal parametric margin support vector classifier and its applications, *Neural Computing & Applications*, 24(3-4):755-764.

Xie, X., Sun, S., (2012). Multitask twin support vector machines, *Neural Information Processing*, *LNCS*, 7664:341-348.

Xu, Y., Wang, L., (2012). A weighted twin support vector regression, *Knowledge-Based Systems*, 33:92-101.

Xu, Y., Yang, Z., Pan, X., (2016). A novel twin support-vector machine with pinball loss. *IEEE Transactions on Neural Networks and Learning Systems*, 28(2): 359-370.

Yan, H., Ye, Q., T, Zhang., Yu, D-J., Yuan, X., Xu, Y., Fu, L., (2018). Least squares twin bounded support vector machines based on L1-norm distance metric for classification. *Pattern Recognition*, 74: 434-447.

Yan, H., Ye, Q-L., Yu, D-J., (2019). Efficient and robust TWSVM classification via a minimum L1-norm distance metric criterion. *Machine Learning*, 108: 993-1018.

Yang L., Dong H., (2018). Support vector machine with truncated pinball loss and its application in pattern recognition. *Chemometrics and Intelligent Laboratory Systems*, 177: 89-99.

Yang, Z. M., Hua, X.Y., Shao, Y.H., Ye, Y.F., (2016). A novel parametric-insensitive nonparallel support vector machine for regression. *Neurocomputing*, 171:649-663.

Ye Q., Zhao C., Gao S., Zhang H., (2012). Weighted twin support vector machines with local information and its application, *Neural Networks*, 35:31-39.

Yildizer, E., Balci, A. M., Hassan, M., & Alhajj, R. (2012). Efficient content-based image retrieval using multiple support vector machines ensemble. *Expert Systems with Applications*, *39*(3), 2385-2396.

Zhao Y., Sun J., (2008) Robust support vector regression in the primal. *Neural Networks*, 21(10):1548–1555.

Zhao Y., Sun J., (2010). Robust truncated support vector regression, Expert *Systems with Applications*, 37(7):5126-5133.

Zhong, P., Xu, Y., Zhao, Y., (2012). Training twin support vector regression via linear programming, *Neural Computing & Applications*, 21:399-407.

Zhu, J., Hoi, S.C., Lyu, M.R.T., (2008). Robust regularized kernel regression. *IEEE transactions on systems, man, and cybernetics, part b (cybernetics)*, 38(6):1639-1644.