Learning Robust Support Vector Machine Classifiers with Uncertain Observations

A THESIS
SUBMITTED FOR THE DEGREE OF
Doctor of Philosophy
IN THE FACULTY OF ENGINEERING

by

Sahely Bhadra

Computer Science and Automation
Indian Institute of Science
BANGALORE – 560 012
MARCH 2012
TO

My Parents
Tapasi and Riten Bhadra
and
My Husband
Mrinal Kanti Das
Acknowledgements

Though my name is printed on the cover of this thesis, the word I does not appear within its chapters. I do this to pay tribute to the myriad contributions of my advisers and collaborators, and the support of my family and friends.

First and foremost I would like to express sincere gratitude and thanks to my adviser, Prof. Chiranjib Bhattacharyya. With his interesting thoughts and ideas, inspiring ideals he made sure I was filled with enthusiasm and interest to do research all through my PhD. He has helped me out in all kind of difficult situations, in which I often found myself, through his insightful comments, suggestions and friendly nature. My current interest and understanding of machine learning is mostly due to him.

I also thank Prof. Aharon Ben-Tal (Technion, Israel), Prof. Arkadi Nemirovski (Technion, Israel), Dr Saketh Nath (IITB, India) and Sourangshu Bhattacharya (Yahoo! Labs, India) for their time, suggestions and valuable comments. I would also like to thank to all the professors whose courses have guided me to solve the problems in my projects.

I thank IBM for appreciating my work by granting me IBM PhD Fellowship Award. I also thank Yahoo! for supporting me financially during the PhD work. I would also like to thank the Indian Institute of Science, India for the graduate scholarship. I would also like to take this opportunity to thank all the people who directly and indirectly helped in finishing my thesis.

IISc is a great home. My stay at the Machine Learning Lab was very joyful, thanks to a constant stream of good friends and colleagues. I enjoyed working with Saketh a lot. I also enjoyed the technical and non-technical discussions with Saketh, Sourangshu, Vinay, Samik, Achintya, Goutham, Adway, Mrinal, Shobhit, and all others.

I am immensely indebted to my parents for their nurture, upbringing and unconditional
love. I am also thankful to my sister, Pratiti, for her moral support and encouragement. Finally, I would like to express my heartfelt gratitude, love and admiration to my husband, Mrinal. Without his incredible support, encouragement, patience and loving kindness, I would not have made it through the program so smoothly.
Publications based on this Thesis


Awards

- IBM PhD Fellowship Award, 2010-11
Abstract

The central theme of the thesis is to study linear and non linear SVM formulations in the presence of uncertain observations. The main contribution of this thesis is to derive robust classifiers from partial knowledge of the underlying uncertainty.

In the case of linear classification, a new bounding scheme based on Bernstein inequality has been proposed, which models interval-valued uncertainty in a less conservative fashion and hence is expected to generalize better than the existing methods. Next, potential of partial information such as bounds on second order moments along with support information has been explored. Bounds on second order moments make the resulting classifiers robust to moment estimation errors.

Uncertainty in the dataset will lead to uncertainty in the kernel matrices. A novel distribution free large deviation inequality has been proposed which handles uncertainty in kernels through co-positive programming in a chance constraint setting. Although such formulations are NP hard, under several cases of interest the problem reduces to a convex program. However, the independence assumption mentioned above, is restrictive and may not always define a valid uncertain kernel. To alleviate this problem an affine set based alternative is proposed and using a robust optimization framework the resultant problem is posed as a minimax problem.

In both the cases of Chance Constraint Program or Robust Optimization (for non-linear SVM), mirror descent algorithm (MDA) like procedures have been applied.
# Contents

Acknowledgements i
Publications based on this Thesis iii
Abstract iv
Keywords xi
Notation and Abbreviations xii

## 1 Introduction
1.1 Introduction ........................................... 1
1.2 Motivation ........................................... 3
1.3 Contribution ......................................... 8
1.4 Overview of Thesis ................................... 11

## 2 Background
2.1 Introduction ........................................... 16
  2.1.1 Structure of the Chapter ......................... 17
2.2 Introduction to Robust Optimization and Chance Constrained Program ........................................... 17
2.3 Support Vector Machine ................................ 21
2.4 SVMs with Robust Optimization ......................... 25
  2.4.1 SVMs and Robust QP ............................... 30
  2.4.2 SVMs and Chance Constraint Program ............... 31

## 3 Robust Classifiers for Interval-Valued Data
3.1 Introduction ........................................... 32
  3.1.1 Review of Past Work ............................... 33
  3.1.2 Contribution ....................................... 35
  3.1.3 Structure of the Chapter ......................... 36
3.2 Robust Classifiers for Interval-Valued Uncertainty .......... 36
  3.2.1 Maximum-Margin Formulation using Bernstein Bounds ........................................... 36
  3.2.2 Geometric Interpretation of the Formulation ....... 40
3.3 Prediction rules and Error metrics ........................ 43
3.4 Experimental Evaluation ................................ 45
## CONTENTS

3.4.1 Datasets ......................................................... 46
3.4.2 Comparison of Margin ......................................... 47
3.4.3 Comparison of Generalization Error ......................... 49
3.5 Summary ............................................................ 50

4 Robustness to Bounded Moments ....................................... 51
4.1 Introduction ......................................................... 51
4.1.1 Review of Past Work ......................................... 53
4.1.2 Contribution ...................................................... 54
4.1.3 Structure of the Chapter ....................................... 55
4.2 Maximum-margin Formulations for Uncertain Data ............... 55
4.2.1 Formulations using Support and Bounds on 2nd Order Moments ......................................................... 56
4.2.2 Formulations using Support and Bounds on Means ............. 63
4.2.3 Note on the Conservative Nature of the Various Formulations ......................................................... 65
4.3 Prediction rules and Error metrics .................................. 66
4.4 Experimental Evaluation ............................................ 69
4.4.1 Datasets ........................................................... 70
4.4.2 Comparison of the Conservative Nature of the Various Robust Constraints ......................................................... 71
4.4.3 Comparison of Classification Margin ............................ 72
4.4.4 Comparison of Generalization Error ............................ 74
4.4.5 Robustness to Moment Estimation Errors ....................... 78
4.5 Summary ............................................................ 79

5 Handling Uncertainty in Kernel Matrices .............................. 81
5.1 Introduction ......................................................... 81
5.1.1 Review of Past Work ......................................... 83
5.1.2 Contribution ...................................................... 83
5.1.3 Organization ...................................................... 84
5.2 Robust Formulations for Handling Uncertainty in Kernel Matrices ......................................................... 84
5.2.1 Uncertainty in Kernel matrix is Gaussian distributed ............ 84
5.2.2 Uncertainty with Finite Support .................................. 86
5.2.3 A Deterministic Optimization Problem .......................... 88
5.3 Algorithms for Solving the Robust Formulation .................... 88
5.4 Prediction rules and Error metrics .................................. 90
5.4.1 Prediction Rules .................................................. 91
5.4.2 Error Metrics ...................................................... 91
5.5 Experimental Evaluation ............................................ 92
5.5.1 Datasets ........................................................... 93
5.5.2 Comparison of Generalization Error ............................ 95
5.5.3 Comparison of Robustness ....................................... 96
5.5.4 Effectiveness of Bound .......................................... 99
5.5.5 Comparison between $RSM$ and $RSM^{(k)}$ ........................ 99
5.5.6 Resolution-aware Protein Structure Classification ................ 102
5.6 Summary ............................................................ 104
# CONTENTS

6 Affine Uncertainty set of Uncertain Kernel 105
   6.1 Introduction ............................................. 105
      6.1.1 Review of Past Work ................................ 106
      6.1.2 Contribution ....................................... 107
      6.1.3 Structure of the Chapter ............................. 107
   6.2 Affine Uncertainty set for Handling Uncertainty in Kernel Matrices 107
      6.2.1 Modeling uncertainty sets for Kernel matrices ........ 109
      6.2.2 Evaluation of base kernels ......................... 111
   6.3 Prediction rules and Error metrics ........................ 112
      6.3.1 Prediction Rules .................................... 112
      6.3.2 Error Metrics ...................................... 113
   6.4 Experimental Evaluation ................................. 113
      6.4.1 Datasets ............................................. 114
      6.4.2 Comparison of Generalization Error ................. 116
      6.4.3 Comparison of Robustness ........................... 117
      6.4.4 Resolution-aware Protein Structure Classification ....... 119
   6.5 Summary .................................................. 121

7 Efficient Algorithm for learning Robust Classifiers 122
   7.1 Introduction ............................................. 122
      7.1.1 Review of Past Work ................................ 123
      7.1.2 Contribution ....................................... 124
      7.1.3 Structure of the Chapter ............................. 125
   7.2 Efficient Algorithm based on Mirror Descent Algorithm (MDA) 125
      7.2.1 Mirror Descent Algorithm ............................ 125
      7.2.2 Solving 7.2 using MDA ................................ 128
      7.2.3 Solving 7.1 using MDA ................................ 131
   7.3 An Algorithm for a special class of Convex-Concave Saddle Point Problem 133
      7.3.1 Assumptions on the Saddle Point Problem ............ 134
      7.3.2 Algorithm for solving Convex-Concave saddle point problem 135
      7.3.3 Application of FSS Algorithm to (7.2) ................. 138
      7.3.4 Application of FSS Algorithm to (7.1) ................. 140
      7.3.5 Discussion ........................................... 142
   7.4 Experimental Evaluation .................................. 142
      7.4.1 Datasets ............................................. 143
      7.4.2 Comparison of Generalization Error ................. 143
      7.4.3 Verification of Convergence of FSS Algorithm .......... 145
      7.4.4 Comparison of Training Time ........................ 145
   7.5 Summary .................................................. 148

8 Conclusion 149
   8.1 Summary ................................................ 149
   8.2 Future Directions ....................................... 151
List of Tables

3.1 Summarizing label prediction methodologies ........................................... 44
3.2 Generalization Performance of IC-SM .................................................. 49
4.1 Summarizing various formulations and corresponding partial information .... 53
4.2 Summarizing label prediction strategies .................................................. 67
4.3 Showing $\varepsilon_{opt}$ for different $\text{OptErr}_{i}^{1-4}$ ................................. 68
4.4 Various relaxations of the chance-constraint ............................................ 72
4.5 Generalization performance of various formulation .................................... 74
4.6 Comparison of deviation in testset error incurred by various robust classifiers 79
5.1 Generalization performance of RSVM .................................................... 96
5.2 Performance of RSVM on protein structure data ....................................... 103
6.1 Various formulations and associated information required. ......................... 114
6.2 Generalization performance of USSVM .................................................. 117
6.3 Performance of USSVM on protein structure data ..................................... 121
7.1 Generalization performance of FSS and MDA based procedure ................. 144
B.1 Table showing values of $\nu$ as a function of $\hat{\mu} \in [0,1]$ at 20 equal increments. 157
B.2 $\nu(\hat{\mu}^{-}, \hat{\mu}^{+}, \hat{\sigma})$ ................................................................. 158
### List of Figures

1.1 (a) Pictorial presentation of $C^\alpha$ atoms of protein d1sra1 (top) and d1gefa1 (bottom). (b) Structural alignment between them. (c) Possible perturbation within resolution limit. (d) Alignment among perturbed structures

2.1 Robust-Optimization Vs Chance-Constrained-Program

2.2 Support Vector Machine

2.3 Robust Classifier for Hyper-Rectangular Uncertainty

2.4 Robust Classifier for Ellipsoidal Uncertainty

3.1 Geometric Interpretation of the Formulation

3.2 A plot of $v(\hat{\mu})$ vs. $\hat{\mu}$

3.3 Geometric interpretation of constraints considered by IC-SM

3.4 Comparison of margins achieved by MM-M, MM-S and IC-SM

4.1 Comparison of the Conservative Nature of the Various Robust Constraints

4.2 Comparison of of Classification Margin of the Various Robust Constraints

5.1 Comparison of robustness for RSVM and Nominal-SVM

5.2 Robustness for RSVM$_{QP}$, RSVM$_{(g)}^{(r)}$ and Nominal-SVM

5.3 Verification of Bound

5.4 Comparison of bound between RSVM and RSVM$_{(g)}^{(r)}$

5.5 Comparison on Robustness of RSVM and RSVM$_{(g)}^{(r)}$

6.1 Comparison of Robustness of USSVM and Nominal-SVM

7.1 Rate of convergence for FSS

7.2 Comparison of training time of MDA, FSS and SOCP
Keywords

Robust Classifier, Interval-Valued Uncertainty, Uncertain in Kernel Matrix, Chance-Constraint-Problem, Robust Optimization, Large Scale Procedure, Machine Learning
Notation and Abbreviations

- $\|\cdot\|_2$ — 2-norm or Euclidean norm of a vector $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
- $\|\cdot\|_p$ — p-norm of a vector $\|\mathbf{x}\|_p = (\sum_{i=1}^{n} x_i^p)^{\frac{1}{p}}$
- $\|\cdot\|_F$ — Frobenius norm of a matrix, $\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} X_{ij}^2}$
- $\text{Tr}(\cdot)$ — Trace of a matrix
- $(A \ast B)$ — Hadamard product of $A, B \in \mathbb{R}^{n \times n}$, with entries $(A \ast B)_{ij} = A_{ij}B_{ij}$
- $(a \ast b)$ — Element-wise product of $a, a \in \mathbb{R}^n$, with entries $(a \ast b)_i = a_ib_i$
- $\mathcal{S}_+^n$ — The space of symmetric positive semi-definite $n \times n$ matrices
- $\lambda_i(\mathbf{Q})$ — The $i^{th}$ eigenvalue of a square matrix $\mathbf{Q}$
- $\lambda_{\max}(\mathbf{Q})$ — The largest eigenvalue of a square matrix $\mathbf{Q}$
- $e_i(\mathbf{Q})$ — The $i^{th}$ eigenvector of a square matrix $\mathbf{Q}$
- $e_{\max}(\mathbf{Q})$ — The largest eigenvector of a square matrix $\mathbf{Q}$
- $\mathbf{1}_E$ — be the indicator function for the event $E$
- $\mathbf{x}$ — Lower case bold-faced symbol represents a vector
- $x$ — Lower case letter represents a scalar
- $\mathbf{x}$ — Lower case tele-type font letter represents a random variable
- pdf — Probability density function
- $\mathbf{x} \sim f_\mathbf{x}$ — The random variable $\mathbf{x}$ has the pdf $f_\mathbf{x}$
- $\mathbf{X}$ — Upper case bold-faced symbol represents a matrix
- $\mathbf{X}_{ij}$ — The element in $i^{th}$ row and $j^{th}$ column of matrix $\mathbf{X}$
- $X$ — Upper case letter represents a constant term
### Notation and Abbreviations

- $X$ — Upper case tele-type font letter represents a random matrix
- $X_{ij}$ — The element in $i^{th}$ row and $j^{th}$ column of random matrix $X$
- $Z$ — Upper case letter represents a random matrix
- CCP — Chance-Constrained Program
- SOCP — Second Order Cone Program
- SVM — Support Vector Machine
- RO — Robust Optimization
- $\mathcal{D}$ — Training dataset
- $n$ — Number of training data points
- $d$ — Dimensionality of training data
- $L$ — Number of uncertain samples for training data points
- $R$ — Number of uncertain samples for test data points
Chapter 1

Introduction

Abstract

This chapter introduces the main theme of the Thesis and discusses the main contributions. The chapter also provides a road-map to the Thesis which serves as a readers guide.

1.1 Introduction

The learning of a binary classifier can be interpreted as constructing a function \( f : \mathcal{X} \rightarrow \mathcal{Y} \), where \( \text{sign}(f(x)) \), \( x \in \mathcal{X} \) gives the class label \( y \in \mathcal{Y} \) of the observation \( x \). Usually, the only information available is a finite set of training examples, known as training datasets \( \mathcal{D} = \{ (x_i, y_i) \mid i = 1, \ldots, n \} \), where \( x_i \in \mathcal{X} \) is a data vector or feature vector representing the \( i^{th} \) data point. In most of the cases \( x_i \) is a \( d \) dimensional vector of real numbers and \( y_i \in \mathcal{Y} (= \{ -1, 1 \}) \) denote class label. According to statistical learning theory, good learning algorithms desire reasonably low generalization error, which is the error in predicting labels of examples not necessarily in \( \mathcal{D} \). Traditional classification algorithm based on the Support Vector Machines (SVMs) \([1]\) can build both linear and non-linear decision boundaries with desirable generalization error, provided the training and the test data points are exactly known. Not only SVM but almost all the existing learning algorithms assume that the data vectors are known accurately and exactly.
But real-world classification data are fraught with uncertainty and noise. Uncertainty in observations can arise due to multitude of reasons, e.g. sampling errors, missing values, resolution of measurement process generating the observations etc. Here uncertainty means that the data vectors are not known exactly, or in other words each observation is a combination of the true value of it and random noise. Hence $i^{th}$ observation can be represented by a random vector $x_i = [x_{i1}, \ldots, x_{id}]$. The bio-medical datasets, gene/protein expression data, protein structure data and image processing data are few examples of noisy data which have already been used in classification and other learning tasks. The effect of even small uncertainty in parameters can be devastating in terms of the quality or feasibility of a solution in an optimization problem. By assuming all observations are accurate, an optimization problem can end up finding a solution which is actually not a feasible solution or at least not a optimum solution while uncertainty is present in the observations. Similarly, for the learning task, over-fitting has long been recognized as a central challenge created by noisy data [1]. As a consequence, learning, optimization and decision-making from data must cope with uncertainty associated with data and must also be well generalized.

The central theme of the thesis is to design classifiers in the presence of uncertain observations. The thesis focuses on binary classifiers but all the results can be applied to multi-class classification problems by “one-vs-all” or “one-vs-one” methods. This Thesis proposes novel methodologies for constructing maximum margin classifiers which are robust to uncertainties in the data points. More precisely, the proposed classifiers should classify every instances of the random vector correctly. For this purpose, one can make some assumptions about the underlying distribution from which these vectors are generated, but generally much information is not available to support these assumptions. The main challenge addressed in this Thesis is in modeling interval-valued uncertainty by not even assuming availability of full second order moment information of uncertain feature vectors. The proposed classifiers only employ partial information like support (bounds on uncertainty of the true data point) and bounds on second order moments (mean and variance) of the uncertain training datapoints. This partial information is either directly available or can easily be derived for most of the noisy datasets.

Robust Optimization (RO) [23] field deals with methodologies for solving several types
Chapter 1. Introduction

of optimization problems with uncertainty. The entity of interest is an uncertain optimization problem where parameters are uncertain but bounded and belong to a given nonempty bounded set. This kind of problems are handled by solving its robust-counterpart — the minimax optimization problem, whose solution is guaranteed to be feasible for any realization of the uncertain parameters and the objective value is no worse than the worst case solution. However, one of the main criticisms received for robust optimization is its inability to incorporate any available distributional information to achieve better performance. On the other hand, in Stochastic Programming [56], one assumes that the data are of stochastic nature with known distribution and seeks a solution which minimizes the expected value of the objective function over the candidate solutions which satisfy the constraints with a given (close to 1) probability. Problems containing this type of probabilistic constraints are known as Chance-Constrained-Programs or CCP. In this thesis RO and CCP methodologies have been applied to build robust classifiers for uncertain datasets.

Recent studies have shown that classifiers which explicitly handle the uncertainty in training data perform better than the classifiers which ignore such information [2, 3, 4]. In the past, robust classifiers which either employ support information [5, 6] or second order moments of the noise distribution [7] were derived. Since these classifiers employ limited partial information i.e. either support or moment information alone, though they achieve robustness to uncertainty, they tend to be overly-conservative. However, uncertainty can be better modeled using richer partial information and that leads to classifiers which are robust but not overly-conservative and hence they achieve better generalization than the existing methods.

1.2 Motivation

This section shows the importance of building classifiers which can explicitly handle the uncertainty in training data. The bio-medical datasets, gene/protein expression data, protein structure data and image processing data are few examples of noisy data which have already been used in classification. In recent times, the application of machine learning algorithms in various fields is increasing rapidly. This compels learning algorithms to learn from various types
of datasets associated with different kinds of noise or uncertainty. Not only that, in most of the cases only little information is available regarding the associated noise or uncertainty. For example:

- **Observation along with minimum/maximum possible values:** In the case of biomedical datasets, the measurement devices vary in terms of resolutions. Lack of complete understanding of the underlying biology further complicates this problem. Moreover in almost all cases, a measurement can vary among different cells within the same tissue of a patient. For example, in the case of cancer diagnosis, a tumorous tissue is analyzed and features are computed for each cell nucleus. However, since the features vary among cells of a tissue, usually, the mean and worst-case (minimum/maximum) feature values of the tissues are considered.

- **A set of values corresponding to one observation:** In the case of gene/protein expression data, uncertainty is inevitable, since experiments are usually noisy and data is prone to be erroneous. So, data for a number of replications of the same experiment are provided.

- **Uncertainty in Kernel matrix due to uncertainty in observation:** A major motivation for our study is the problem of automated protein structure classification. Classification of protein structures into various classes like families, super families etc remains an important research challenge in computational biology (see [17]). Kernel based classifiers are becoming increasingly popular [15, 16], for addressing this problem.

Usually a protein structure is specified by the positions of $C^\alpha$ atoms. In the sequel we will denote each protein structure by a set

$$P = \{ \mathbf{c}_i \in \mathbb{R}^3 | i = 1, \ldots, s \}. \quad (1.1)$$

---

1 Examples are the Wisconsin breast cancer diagnostic/prognostic datasets available at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)

2 Examples are the Micro-array data of various drugs available at http://www.ncbi.nlm.nih.gov/geo/ with accession number GSE2187.
where each $C^\alpha$ atom is determined by spatial coordinates $\mathbf{c}_i = \{c_{i1}, c_{i2}, c_{i3}\}$ obtained by X-ray crystallography. Automated classification of such structures is an extremely useful and challenging problem in computational biology. In the absence of any natural vector representation for such data, construction of classifiers remain a challenge. In the recent past kernel based methods [15, 16] have emerged as an interesting alternative to this problem.

Biologists often compute the similarity between a pair of structures by first computing an alignment and then measuring the quality of alignment by a suitable measure such as root mean square deviation (RMSD). We do not formally define the notion of alignment and RMSD in this paper but the refer the interested reader to [18, 17] Though computing structural alignment is an intractable problem there are several approximate algorithms like DALI [17], CE [18] etc, which works well in practice. Existing literature [15, 16] on kernel design for such problems rely on such structural alignment programs.

All such procedures implicitly assume that the protein structures are determined exactly, i.e. the location of the atoms constituting the structure is known precisely. Unfortunately in reality, the coordinates, $\mathbf{c}_i$, are difficult to determine with exact precision, which is highly dependent on the resolution of X-ray diffraction experiment $^3$. For a protein structure $P$, the resolution information $r$, specifies the error in each coordinate. More precisely the position of the $i$th atom in a protein structure $P$ (see (1.1)) could be anywhere in the uncertainty box $\{\mathbf{z}|\|\mathbf{z} - \mathbf{c}_i\|_\infty \leq r\}$. Incorporating resolution information can thus be understood as representing each protein structure as follows: For a set $P$, as defined in (1.1), there is an associated uncertainty set

$$U(P) = \{R|R = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\} \ | \ |\mathbf{z}_i - \mathbf{c}_i\| \leq r; \ i = 1, \ldots, n\}$$

(1.2)

where $r > 0$, is given. We would refer to $P$ as the nominal structure and $U(P)$ as the uncertainty set associated with it. The set $U(P)$ characterizes all alternative structures, including $P$, at a particular resolution $r$.

$^3$http://www.rcsb.org/pdb/
Comparison of two proteins given access to only $U(P)$ is an extremely difficult problem. Even when $r$ is small, the alignment scores between two nominal structures, $P_i$ and $P_j$ can differ significantly from the alignment scores between $P'_i$ and $P'_j$ where $P'_i \in U(P_i), P'_j \in U(P_j)$. This difference in alignment scores can lead to different kernel values.

For example, consider two proteins $^{4}$ d1vsra1 (denote it by P) and d1gefa1 (denote it by P’) belonging to protein superfamily Restriction endonuclease-like. The structures for these have been determined at resolutions 1.8 Å and 2.0 Å respectively. The program DALI computes a structural alignment with RMSD of 3.7 Å between these two structures. Figure 1.1(a) shows pictorial presentation of $C^\alpha$ atoms of these two proteins while Figure 1.1(b) shows structural alignment between them. If one ignores the uncertainty one obtains a kernel value of 1.3585, using the kernel function described in \cite{16}. On randomly generating alternate structures by sampling from the corresponding uncertainty box (1.2) we observe that the kernel value ranged from $1.1542(=K_{\text{min}}) \leq K(P_i,P_j) \leq 1.4964(=K_{\text{max}})$, see Figure 1.1(c,d). This variability is substantial. Indeed majority of the kernel values between any pair of nominal protein

$^{4}$One should refer to them as SCOP domains. But to lighten the discussion on the biology side we refer to them as proteins
structures, more than 60%, in superfamily **Restriction endonuclease-like** lie between $K_{\text{min}}$ and $K_{\text{max}}$.

This demonstrates that accounting for resolution information leads to uncertainty in kernel values, which cannot be ignored.

- Other than these, image classification and automated call-routing are also examples of applications where the data is prone to be erroneous.

All the examples stated above have different kind of information available. For example, cancer diagnosis data has both statistical (First order moment or mean of every feature) and support information (minimum and maximum values) for every features, while microarray data has 5 readings per feature value from which one can calculate statistical information or approximated moments, like sample mean, sample variance and also approximated support. On the other hand, protein structure data provides a boundary around possible position of each atom, which in-turn provides some approximated boundary for kernel values between 2 proteins. To get robust classifiers for all these data sets, handling uncertainties by only knowing partial information regarding uncertainties is required. Not only that, it would be better if the classifiers can be robust to moment estimation errors too. Because, in most of the cases, sample mean or sample variance represents the moments which are not accurate. Moreover, all the popular kernel functions for protein structure data are very complex and non-trivial. In order to get good non-linear classifiers, not only for protein structure data, but also for various other datasets, very complex kernel functions are being used recently. Therefore, to build robust classifiers for these kind of datasets, it is required to have robust non-linear classifiers which can handle uncertainty in any arbitrary kernel matrix. This Thesis has tried to solve these problems. Above all, Thesis also focused on the fact that, derived robust classifiers must be well generalized and of course scalable to get maximum use of it.
1.3 Contribution

The main contribution of this Thesis is to derive robust classifiers from partial knowledge of the underlying uncertainties both in case of linear and non-linear SVM formulations. Thesis presented a few novel methodologies which can handle following type of uncertainty in observation.

- **Interval valued uncertainty:** We assume bounds of uncertainty in every feature value of a feature vector is known along with some information of bound of its moments.

- **Gaussian uncertainty in kernel matrix:** We assume each element of kernel matrix is uncertainty and underlying distribution is Gaussian with known mean and variance.

- **Interval valued uncertainty in kernel matrix:** We assume each element of kernel matrix is uncertainty and underlying distributions are unknown but noises are bounded and bounds are known.

- **Affine set uncertainty in kernel matrix:** Given a nominal kernel matrix $\mathbf{K}$ and few other kernel matrix $\mathbf{K}_l$, $\forall l = 1, \ldots, L$ uncertainty in kernel matrix can be interpreted by a set of all possible affine combination of them $\mathcal{U}(\kappa) = \{ \mathbf{K} = \mathbf{K} + \sum_{l=1}^{L} \eta_l \mathbf{K}_l, \| \eta \|_p \leq \kappa, \mathbf{K}_l, \mathbf{K} \in \mathcal{A}^n \} (p \geq 1)$

This Thesis presents a novel maximum-margin classifier which achieves good generalization on data having interval-valued uncertainty. The key idea has been to employ chance-constraints in order to handle uncertainty and then approximate it by a tractable SOCP formulation using Bernstein schemes [10, 11]. To the best of knowledge acquired, this is the first time Bernstein approximation schemes are employed for approximately relaxing linear chance constraints in CCP based formulations. Using similar idea, a maximum margin classifier was derived which uses support information of data along with support information of the first and the second order moments rather than the exact moments. Thus, in addition of being robust to uncertainty and not being overly conservative, the proposed classifiers are also inherently robust to moment estimation errors. This is the first time in the literature, a robust classifier has been proposed which can handle uncertainty in kernel matrices and is readily applicable
to arbitrary kernel functions. Both CCP and robust optimization based approaches have been studied here. Moreover, efficient algorithms along with special solvers have been proposed for learning resultant classifiers. Details of contributions are as follows:

**Partial information of both support and statistics:** The key novelty in this Thesis is that, to handle interval-valued uncertainty it requires only partial information of support and moments, i.e, mean and variance of uncertain examples alone and the approach makes no assumptions regarding full co-variance of uncertainties in various feature values. The proposed methodologies also explore potential of partial information such as bounds on second order moments along with support information. In the past, robust classifiers which either employ support information \([5, 6]\) or second order moments of the noise distribution \([7]\) tend to be overly-conservative. However, as richer partial information is employed here, uncertainty is better modeled — leading to classifiers which are robust but not overly-conservative. Since the proposed classifiers employ richer partial information and model the uncertainty in a better way, they achieve better generalization than the existing methods.

**Bernstein like bounding scheme — less conservative:** In case of linear classification, a new bounding scheme based on Bernstein inequality has been proposed, which models interval-valued uncertainty in a less conservative fashion and hence are expected to generalize better than the existing methods. The idea is to employ chance-constraints which ensure that the uncertain data points are classified correctly with high probability. Optimization problems involving chance constraints often turn out to be NP-hard. Until now, only the Chebyshev bounding schemes were employed to relax various CCP based learning formulations \([8, 7, 9]\). In this Thesis, Bernstein bounding schemes have been employed \([10, 11]\) for relaxing the resulting Chance-Constrained Program (CCP) as a Second Order Cone Program (SOCP), which can be solved using interior point solvers \([12]\).

**Robust to moment estimation error:** By employing Bernstein bounding schemes to relax the resulting chance-constrained program as a convex second-order-cone-program, the proposed methodologies also explore potential of partial information such as bounds on second
order moments along with support information and turn out to be further less conservative. In addition to that, bounds on second order moments make the resulting classifiers robust to moment estimation errors. It achieves wider classification margins and hence better generalization than the existing methods.

**Uncertainty in arbitrary kernel:** The kernelized SVM [1] was introduced to build non-linear decision boundaries or classifiers by applying the kernel trick (originally proposed by [13]). During the training phase, instead of using data vectors, the kernelized SVM deals with kernel matrix \( \mathbf{K} \) defined on the training data set. The kernel matrix, \( \mathbf{K} \), is a \( n \times n \) matrix, where \( K_{ij} \) can be understood as dot product between implicitly defined feature map over examples \((x_i, x_j) \in \mathcal{D}\). Uncertainty in the dataset will lead to uncertainty in the kernel matrix \( \mathbf{K} \). Uncertainty in kernel can originate from various other reasons in addition to the noisy data, for example Hash Kernel [14]. In this Thesis, the problem of uncertainty in a kernel matrix in the SVM framework is studied to build non-linear robust classifiers. This is a very novel attempt for building robust classifiers which can handle uncertainty in any arbitrary kernel matrix. It discusses a chance-constraint based setup when the uncertainty in the kernel matrix is modeled as independent additive noise \( Z_{ij} \) over the kernel entries \( \mathbf{K}_{ij} \). Hence, observed kernel matrix \( \mathbf{K} = \mathbf{\bar{K}} + Z \), where \( \mathbf{\bar{K}} \) is a specific kernel matrix and \( Z \) is noise matrix. A novel distribution free large deviation inequality is derived to get an upper bound on the trace of a random matrix and when this inequality is applied to the problem at hand, it leads to a co-positive program. Although such formulations are NP hard, under several cases of interest the problem reduces to convex programs.

However, the independence assumption is restrictive and may not always lead to valid uncertain kernels. To alleviate this problem an affine set based alternative is also proposed which models only valid uncertain kernels. Using a robust optimization framework the resultant problem is posed as a minimax problem. This formulation here has been applied on the same problem as described in the previous paragraphs. Since this formulation models uncertainty in kernels in a better and valid way, it achieves better generalization.
Scalable algorithms: Another key contribution of this Thesis is to provide efficient training algorithms for building robust classifiers. In this Thesis various methods have been discussed to build robust classifiers which can classify uncertain data more accurately. But almost all of them turn out to be solving a conic convex program more precisely Second Order Cone Program or SOCP. Though many Interior Point (IP) method based solvers are available to solve SOCP but complexity of them are around $O(n^4)$. As a consequence, to learn a robust classifier using only a few thousands of training data points, even may need few days and moreover, most of the time, it ends up in crashing the system. Due to the increasing interest in handling uncertain data, the need for fast and efficient training algorithms in the domains of uncertain data classification increases. In this Thesis, for both CCP and RO approaches, training has been interpreted as a minimax problem, where the minimization problem is a minimization of a convex function over a convex set and the maximization problem can be be solved using an SVM. These minimax problems have been solved efficiently for large scale data by proposing an oracle type Mirror Descent Algorithm (MDA) based procedure. Moreover, for both the cases of Chance Constraint Program or Robust Optimization(for non-linear SVM), a novel saddle point based procedure has been applied. This novel algorithm uses only first order information, yet proved to decrease the initial error by a factor of $O(1/T^2)$, after $T$ iterations. This makes both the formulations scalable.

Performance measurements: Methodologies for classifying uncertain test datapoints along with uncertain kernel and error measures for evaluating robustness of classifiers have been discussed thoroughly too.

1.4 Overview of Thesis

This section shows the organization of remainder of the Thesis and serves as a road-map to the reader.

This Thesis broadly consists of three main parts: Motivation and background study (Chapters 1 and 2), Contribution (Chapters 3, 4, 5, 6 and 7) and Conclusion (Chapter 8). Chapter 1
sketches an overview of the Thesis and gives motivation for the problems attempted in the Thesis. It also gives a brief overview techniques used and also contributions of Thesis.

Chapter 2 gives a brief introduction to Robust Optimization and Chance Constraint Programming. It also reviews existing literature related to building robust classifiers based on SVM using results out of various researches in robust optimization field.

Chapter 3 presents a novel methodology for constructing maximum-margin classifiers which are robust to interval-valued uncertainty in data. The idea is to employ chance-constraints which ensure that the uncertain data points are classified correctly with high probability. Section 3.1 reviews past work and discusses issues with the existing methods. The key novelty is in employing Bernstein bounding schemes to relax the resulting chance-constrained program as a convex second-order-cone program has been represented in Section 3.2. Section 3.2.2 gives a geometrical interpretation for the proposed formulation. Section 3.3 presents the methodology for labeling interval-valued test examples and discusses error measures for evaluating the performance of interval data classifiers. Experimental results on synthetic and real-world datasets, presented in Section 3.4 show that the proposed classifiers are better equipped to handle interval-valued uncertainty than state-of-the-art. Section 3.4.2 compares classification margins ($2/\|w\|_2$) achieved by proposed IC-SM and existing MM-S, which represents state-of-the-art and the proposed interval data classifiers respectively. The key results of this chapter are presented in Section 3.4.3. These results compare the generalization performance of proposed robust formulation with various existing classifiers. Section 3.5 summarizes the work and concludes the chapter.

Chapter 4 derives tractable maximum-margin formulations which employ both the support and the second order moment information of the uncertain datapoints in order to build the linear decision boundary. Moreover, the proposed classifiers require the knowledge of bounds on second order moments rather than the exact moments, which are often unknown. Thus, in addition of being robust to uncertainty and not being overly conservative, the proposed classifiers are also inherently robust to moment estimation errors. Section 4.2 presents the main contribution of the chapter, a maximum-margin SOCP formulation which employs the support and bounds on the second order moments of the uncertain datapoints in order to achieve robustness.
Chapter 1. Introduction

The section also presents various specializations of this formulations. Section 4.3 discusses the issue of classifying uncertain test datapoints and presents various error measures which evaluate the performance of classifiers which handle uncertain data. In Section 4.4, experimental results which compare the performance of the proposed methods and the existing methods are presented. The first set of experiments, presented in Section 4.4.2, compare the conservative nature of Chebyshev and Bernstein bounding schemes for relaxing CCP based learning formulations. Section 4.4.3 presents experiments which compare the margin, $2/\|w\|_2$, achieved by the proposed and existing robust classifiers on synthetic datasets. Section 4.4.4 presents the key empirical results of the chapter — comparison of generalization performance of various robust classifiers discussed in this chapter. As mentioned earlier, classifiers derived using Bernstein relaxation schemes are also inherently robust to moment estimation errors. This is because the proposed classifiers require knowledge of moment bounds rather than the exact moments themselves. Section 4.4.5 presents experiments comparing the robustness of various uncertain data classifiers to moment estimation errors. The chapter concludes in Section 4.5, by summarizing the work.

Chapter 5 explores Chance-Constrained program for designing kernelized classifiers when observations are uncertain. More precisely, it is about the problem of designing robust classifiers when there is uncertainty in the kernel matrix. The impact of uncertainty in individual examples is treated as an additive uncertainty. The main contributions are described in section 5.2. Following two cases are studied here, a.) uncertainty follows Gaussian distribution, and b.) it has finite support. For case of Gaussian distribution a novel formulation is derived in Section 5.2.1. A major contribution of this chapter is a novel large deviation inequality which applies to the finite support case and the Section 5.2.2 presents that. Section 5.3 presents algorithms to solve such problems, and Section 5.4.2 discusses metrics for measuring the performance of resultant classifiers. Section 5.5 reports experimental results. Section 5.5.2 presents the key empirical results of the chapter — comparison of generalization performance of various robust classifiers discussed in this chapter on synthetic data. Section 5.5.3 presents experiments which compare the robustness achieved by the proposed and existing robust classifiers on synthetic datasets. In Section 5.5.4, the effectiveness of bounds claimed in main
Chapter 1. Introduction

Theorem of the chapter is experimentally verified. Experimental results which compare accuracy and robustness of the proposed **RSVM**, with state of the art methods for protein structure classification are presented in Section 5.5.6. Section 5.6 summarizes the work and concludes the chapter.

Chapter 6 solves the same problem of designing classifiers when the Kernel matrix is uncertain but considers the fact that any uncertain kernel is a valid one. This is because of the fact that, CCP based approaches was unable to model symmetricity and positive-semi-definite property of kernel matrices. Hence modeling the uncertainty in kernels were inappropriate. In Chapter 6 Thesis defines an affine uncertainty set of uncertain kernel which leads to a valid model of uncertainty over kernel matrices. Using a robust optimization based framework, the problem of designing the robust classifiers was posed as a minimax problem in Section 6.2. The minimax procedure is solved as a second-order-conic convex program. In Section 6.3 the ground for a comprehensive computational study by introducing various prediction rules and related error metrics is prepared. The results of the computational study are described in Section 6.4. It begins with a brief description of datasets in Section 6.4.1. The first issue is discussed in Section 6.4.2 is generalization performance of proposed classifiers in compare to state-of-art. A comparative study of robustness is presented in Section 6.4.3. The chapter concludes in Section 6.5, by summarizing the work.

Chapter 7 rewrites CCP based **RSVM** formulation as minimax problem when $\beta$ is positive definite matrix. Therefore, for both CCP and RO approaches training can be interpreted as minimax problem where the minimization problem is a minimization of a convex function over a convex set and the maximization problem can be be solved as a SVM. Section 7.2 solves these minimax problems efficiently for large scale data by proposing an oracle type Mirror Descent Algorithm (MDA) [19, 20] based procedure. Moreover, Section 7.3 presents another efficient procedure based on a saddle point algorithm, named as FSS and discuss its applicability to both the minimax problems. This novel algorithm uses only first order information, yet proved to decrease the initial error by a factor of $O(1/T^2)$, after $T$ iterations. The results of the computational study are described in Section 7.4. The section is organized as
follows. The section begins by a discussion of generalization performance of efficient algorithms in comparison with their SOCP counterparts, which is present in Section 7.4.2. Section 7.4.3 discusses the experimental verification of the convergence rate of FSS procedure. Next the scalability of MDA and FSS procedure is discussed in Section 7.4.4. Finally, Section 7.5 summarizes the work and concludes the chapter.

Chapter 8 concludes the Thesis by summarizing main contributions. The chapter also discusses directions for future work.
Chapter 2

Background

Abstract

This chapter provides the background necessary for subsequent developments in this thesis. The thesis mainly talks about formulating few novel robust classifiers built on Support Vector Machine. Therefore, a small introduction to Support Vector Machine along with an introduction to Robust Optimization and Chance-Constraint-Programming is given here. This chapter has also tried to recall few important existing algorithmic and learning based techniques which apply robust optimization to build robust classifiers based on Support Vector Machine to handle uncertain training data.

2.1 Introduction

Support Vector Machine (SVM) is one of the most popular classification algorithm in machine learning literature. It can build both linear and non-linear decision boundaries with desirable generalization error, provided the training and the test data points are exactly known. Learning algorithm for SVM is mainly about solving a quadratic optimization problem. Optimization algorithms affected by parameter uncertainty has long been a focus of the mathematical programming community. Solutions to optimization problems can exhibit remarkable sensitivity to perturbations in the problem parameters, thus often rendering a computed solution highly infeasible, suboptimal, or both [21]. These issues have already been studied in related fields, particularly in the field of robust control and are known as Robust-Optimization in the community (please see the survey [22] and the textbook [23] and references to the original papers.
Robust Optimization is built on the premise that the parameters vary arbitrarily in some apriory known bounded set, called the uncertainty set. In contrast, Stochastic Programming also deals with optimization with uncertain parameters (e.g., [24, 25]), but assumes the uncertainty has a probabilistic description. The Chance-Constraint-Program is a well known branch of Stochastic Programming, where constraints are probabilistic.

On the other hand, machine learning field has also started its exploration in building classification models on uncertain data. Robust versions of binary classification problems are explored in several papers. While [2, 29] studied the classification of uncertain data using the support vector model, [8] constructed Minimax Probability Machine (MPM), a classifier by maximizing the probability of correct classification of unseen data points by using minimax approach.

2.1.1 Structure of the Chapter

The remainder of this chapter is organized as follows: Section 2.3 provides a brief introduction to Support Vector Machine. Section 2.2 provides a brief introduction to Robust Optimization along with a short introduction of Chance-Constrained Program. Section 2.4 studies application of Robust Optimization techniques in Support Vector Machine along with a review of few existing robust classifiers based on SVM mainly to handle uncertainty in training data.

2.2 Introduction to Robust Optimization and Chance Constrained Program

In recent years, robust optimization has gained substantial popularity as a competing methodology for solving several types of optimization problem with uncertainty. The entity of interest
is an uncertain optimization problem where \( f, g_i : \mathcal{P} \subseteq \mathbb{R}^n \Rightarrow \mathbb{R} \)

\[
\text{(UOP)} \quad \min_{x \in \mathbb{R}^n} f(x, \Psi)
\]

\[g_i(x, \Psi) \leq 0, \quad \forall \Psi \in \mathcal{U}, \quad i = 1, \ldots, m \tag{2.1}\]

where \( x \in \mathbb{R}^n \) is the vector of decision variables, \( \Psi \in \mathbb{R}^k \) is a vector of uncertain parameters and \( \mathcal{U} \subseteq \mathbb{R}^k \) is a given non empty set. The (UOP) is in fact a family of problems - one for each realization of \( \Psi \). Uncertainty means that the data vector \( \Psi \) is not known exactly at the time when the solution has to be determined. As a result, it is unclear what does it mean to solve an uncertain problem. In Robust Optimization (RO), the data is assumed to be uncertain but bounded, that is, varying in a given uncertainty set \( \mathcal{U} \), and the aim is to choose the best solution among those immunized against data uncertainty. A candidate solution to (2.1) is called as immunized against uncertainty if it remains feasible for all realizations of the data \( \Psi \) from the uncertainty set \( \mathcal{U} \).

In the RO framework the family of problems, (UOP) is replaced by its robust counterpart (RC) the semi-infinite problem:

\[
\text{(RC)} \quad \text{sol}^* = \min_{x} \max_{\Psi \in \mathcal{U}} f(x, \Psi)
\]

\[g_i(x, \Psi) \leq 0, \quad \forall \Psi \in \mathcal{U}, \quad i = 1, \ldots, m \tag{2.2}\]

A solution of (RC) is guaranteed to be feasible to (UOP) for any realization of \( \Psi \in \mathcal{U} \) and the objective function is no worse than \( \text{sol}^* \). Hence \( \text{sol}^* \) are interpreted as the best immunized against uncertainty solutions of an uncertain optimization problem (UOP). The outlined approach originates in [48].

The major challenges associated with the Robust Optimization methodology are:

- In general the (RC) of an (UOP) may have infinite number of constraints and is often NP hard.

- When and how can one reformulate (2.2) as a computationally tractable optimization problem, or can at least approximate (2.2) by a tractable problem.
• How to specify reasonable uncertainty sets $\mathcal{U}$ in specific applications.

However, for some suitable uncertainty set, the robust counterparts of some important generic convex optimization problems (linear optimization (LP), second order cone optimization problems (SOCP), semidefinite optimization (SDP)) are either exactly or approximately equivalent to tractable problems, which are efficiently solvable via interior point methods. Elaborate discussion on uncertainty sets and computationally attractive corresponding robust optimization methodologies has already been done by Ben-Tal and Nemirovski [49, 21, 11], El-Ghaoui et al. [30, 50, 8], Iyangar and Goldfarb [51, 52] and Bertsimas [53, 54, 22]. A few interesting results of the Robust Optimization methodology in the cases of uncertain Linear, Conic Quadratic and Semidefinite Programming has been reviewed in Appendix A.

**Chance Constrained Program:** One of the main criticisms of robust optimization is its inability to incorporate any available distributional information to achieve better performance. Available distributional information is used in the field of Stochastic Programming [56]. In Stochastic Programming — historically, the first methodology for handling data uncertainty in optimization — one assumes that the data are of stochastic nature with known distribution and seeks for a solution which minimizes the expected value of the objective over candidate solutions which satisfy the constraints with a given (close to 1) probability. Problem containing this type of probabilistic constraint is known as *Chance-Constrained-Program* or CCP.

![Figure 2.1: Linear classifier(left) and robust linear classifier using Robust-Optimization(middle) and Chance-Constrained-Program(right)](image)

Instead of emphasizing on decision within feasible region, Chance Constrained Programming insist on decisions guaranteeing feasibility *as much as possible*. This loose term refers
once more to the fact that in real world situation constraint violation can almost never be avoided because of unexpected extreme events. On the other hand, when knowing or approximating the distribution of the random parameter, it makes sense to call decisions feasible (in a stochastic sense) whenever they are feasible with high probability, i.e., only a low percentage \((0 \leq \epsilon << 1)\) of realizations of the random parameter leads to constraint violation under this fixed decision. A generic way to express constraint in (UOP)(2.1) using such a probabilistic or chance constraint is as following inequality:

\[
\text{Prob}(g_{i}(x, \Psi) \leq 0, \forall i = 1, \ldots, m) \geq 1 - \epsilon
\]

Where \(\text{Prob}\) is a probability measure, \(\Psi\) is a random vector with pdf \(f_{\Psi}\) and \(0 \leq \epsilon << 1\). This is an example of joint chance constraint.

Sometimes, the probability level \((1 - \epsilon)\) is strictly fixed from the very beginning (e.g., 0.95, 0.99 etc.). In other situations, the decision maker may only have a vague idea of a properly chosen level. Of course, it is a fact that, lower values of \(\epsilon\) lead to fewer feasible decisions on \(x\) in (2.3), hence progresses to optimal solutions at higher costs. Fortunately, it turns out that usually \(\epsilon\) can be decreased over quite a wide range without affecting too much the optimal value of some problem, until it closely approaches 0 and then a strong increase of costs is observed. In this way, models with chance constraints can also give a hint to a good compromise between costs and safety. Figure 2.1 shows that margin (hence generalization) of linear classifier increases with help of chance-constraint setting in comparison to that with Robust Optimization setting.

For individual constrains assumption it becomes:

\[
\text{Prob}(g_{i}(x, \Psi) \leq 0) \geq 1 - \epsilon_i, \quad \forall \ i = 1, \ldots, m
\]

Hence resultant chance-constraint-program (2.1) turns out to be:

\[
\text{(CCP)} \quad \min_{x \in \mathbb{R}^n} f(x, \Psi)

\text{Prob}(g_{i}(x, \Psi) \leq 0) \geq 1 - \epsilon, \quad \Psi \sim f_{\Psi}, \quad i = 1, \ldots, m
\]
When $\Phi_f(x) = \text{Prob}(g_i(x, \Psi) \leq 0)$ is known, a tractable deterministic counter part for (CCP) is:

$$
\min_{x \in \mathbb{R}^n} f(x, \Psi)
\Phi_i(x) \geq 1 - \varepsilon_i, \quad i = 1, \ldots, m
$$

(2.4)

For example, if $g(x, \Psi) = h(x) - \Psi$, then $\text{Prob}(g(x, \Psi) \leq 0)$ is equivalent to $\Phi(x) = F_\Psi(h(x))$. The main job in chance constraint programming is to numerically find out corresponding deterministic function $\Phi$ and its derivatives.

Despite of all challenges and difficulties in both chance constraint program and robust optimization based methods, extensive research on the subjects in the recent years was also aimed at the applications of the methodology in various areas including Portfolio selection [52, 57, 58], Machine Learning [30, 50, 8, 2, 29, 7], Control [59, 60], Inventory Theory [61], Signal processing and estimation, circuit design etc. In machine learning field, RO-based learning algorithms have been proposed for classification [8, 2, 29]. This Thesis also explores applicability of RO and CCP in building robust classifiers to handle interval valued data uncertainty by using partial information of support and statistics of underlying distribution. Prior to that, an overview of SVM and it’s existing robust variation is given in following section.

### 2.3 Support Vector Machine

The Support Vector Machines (SVMs) algorithm developed by Vapnik [1] is based on statistical learning theory. It continues to be one of the most successful algorithm for classification. SVMs address the classification problem by finding the hyperplane in the feature space that achieves maximum sample margin when the training samples are separable. Maximum margin can be achieved by minimizing the norm of the classifier. For example, Figure 2.2 shows two groups of data and a separating hyperplane with maximum margin. Such a hyperplane with maximum margin is likely to be generalized better and are supposed to classify “unseen” or testing data points correctly.
Linear SVMs

Linear SVMs are the simplest form of SVMs and is applicable in case of linearly separable data points. They actually try to find a hyperplane \( w^\top x + b = 0 \) which separates two classes of training data from each other by satisfying the constraints \( w^\top x_i + b \geq 1, \ \forall \ i \text{ where } y_i = 1 \) and \( w^\top x_i + b \leq -1, \ \forall \ i \text{ where } y_i = -1 \). These can be combined into one set of inequalities:

\[
y_i(w^\top x_i + b) \geq 1 \ \forall \ i = \{1, \ldots, n\}.
\]

The distance between two hyperplanes \( (w^\top x_i + b = 1 \text{ and } w^\top x_i + b = -1) \) is \( \frac{2}{\|w\|} \) and this is known as margin of the classifier. Hence optimization problem which maximizes the margin along with satisfying the above constraints is:

\[
\min_{w, b} \quad \frac{1}{2} \|w\|^2
\]

s.t. \( y_i(w^\top x_i + b) \geq 1, \ i = 1, \ldots, n \) \hspace{1cm} (2.5)
The decision function is \( f(x) = \text{sign}(w^T x + b) \). However, the optimization problem (2.5) will not have a solution if \( \emptyset \) is not linearly separable. To deal with such cases, soft margin SVM has been introduced. It allows mislabeled data points while still maximizing the margin. The method introduces slack variables, \( \xi_i \), which measure the degree of misclassification. The following is the optimization problem for soft margin SVM.

\[
\begin{align*}
\min_{w, b, \xi_i} & \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \quad y_i(w^T x + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n
\end{align*}
\]

Here \( C \) is a user-given regularization parameter that determines the tradeoff between the margin size and the amount of error in training.

**Kernelized SVM** If the training data is not linearly separable but separable by some non-linear function, linear SVMs must be extended to nonlinear SVMs in order to learn non-linear separable function. The process of finding classification functions in this case consists of two steps. First, the input vectors are transformed into high-dimensional feature vectors where the training data can be linearly separated. Then, SVMs are used to find the hyperplane of maximal margin in the new feature space. The separating hyperplane becomes a linear function in the transformed feature space but a nonlinear function in the original input space. Let \( x \) be a vector in the \( d \)-dimensional input space and \( \phi(\cdot) \) be a nonlinear mapping function from the input space to the high-dimensional feature space. The hyperplane representing the decision boundary in the feature space is defined as \( w^T \phi(x) + b = 0 \). When considering the dual form of SVM using \( \phi(\cdot) \) optimization problem is:

\[
\begin{align*}
\max_{\alpha} & \quad \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\
\text{s.t.} & \quad \sum_{i=1}^{n} \alpha_i y_i \\
& \quad 0 \leq \alpha_i \leq C \quad \forall \ i = 1, \ldots, n
\end{align*}
\]

(2.7)
Using the $\phi(\cdot)$ function, the weight becomes $w = \sum_{i=1}^{n} \alpha_i y_i \phi(x_i)$ and the decision function becomes $\text{sign}(\sum_{i=1}^{n} \alpha_i y_i \phi(x_i)^T \phi(x) + b)$. Note that the feature mapping functions in the optimization problem and also in the classifying function always appear as dot products between pairs of vectors in the transformed feature space. Computing the inner products in the transformed feature space seems to be quite complex and suffer from the curse of dimensionality problem. To avoid this problem, the kernel trick is used. The kernel trick [26] replaces the inner product in the feature space with a kernel function $K$ in the original input space as $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$. A kernel function is valid if the gram matrix $K$ is positive-semi-definite for all finite set of data points. Then the optimization function turns out to be

$$\max_{\alpha \in S_n, t_1} \alpha^T e - \frac{1}{2} t_1 \quad \text{s.t.} \quad \alpha^T Y K Y \alpha \leq t_1$$

where $S_n = \{ \alpha | 0 \leq \alpha_i \leq C, \sum_{i=1}^{n} \alpha_i y_i = 0 \}$ and $Y = \text{diag}(y_i)$. Each entry of the matrix $K$, is defined by $K_{ij} = K(x_i, x_j)$. The decision function is

$$f(x) = \text{sign} \left( \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b \right).$$

Kernel based classifiers are extremely powerful as they can define complicated decision boundaries via the kernel functions and are becoming increasingly popular.

**Generalization Performance** The key issue in the task of learning from a set of observed samples is the estimation of the risk (i.e., generalization error) of learning algorithms. Typically, its empirical measurement (i.e., training error) provides an optimistically biased estimation, especially when the number of training samples is small. Assuming that, the dataset $\mathcal{D}$ consists of $n$ i.i.d. samples with underlying probability distribution $\text{Prob}(x, y)$ and the loss function is $l(\cdot, \cdot)$, the risk can be defined as expected loss:

$$\text{Risk} = E_{\text{Prob}(x,y)} l(f, y).$$
While the empirical loss is:

$$EmpiricalLoss = \frac{1}{n} \sum_{\mathcal{X} \in \mathcal{X}} l(f, y_i).$$

Minimizing the $EmpiricalLoss$ alone can lead to poor classification performance (i.e. higher Risk) for new unlabeled data due to overfitting [1]. To get better generalization performance deviation of Risk from $EmpiricalLoss$ should be bounded. The most popular bound based on uniform convergence considers $l(f(x_i), y_i) = \frac{1}{2} |y_i - f(x_i)|$ and states that with probability $1 - \varepsilon$ for some $0 \leq \varepsilon \leq 1$ following holds.

$$Risk \leq EmpiricalLoss + \sqrt{\frac{h_{vc}(\log(2n/h_{vc}) + 1) - \log(\varepsilon / 4)}{n}}$$

(2.10)

Where $h_{vc}$ is known as VC-dimension of function class and can be minimized by “structural risk minimization”. One of the most popular methods to handle this problem is to minimize the regularization term along with the training error. Regularization is typically expressed as norm of the classifier.

Both the formulations of linear and non-linear SVM consider $x_i$’s and $y_i$’s are deterministic and accurate. While, in reality, both $x_i$’s and $y_i$’s can be corrupted or uncertain. When any of them is uncertain, then SVM can be solved with the help of Robust Optimization methodologies.

### 2.4 SVMs with Robust Optimization

The Optimization problem for linear SVMs can be rewritten as

$$\min_{w, b, \xi} \quad Reg(w, b) + C \sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n$$

(2.11)

where $Reg(w, b)$ is a regularization term (e.g., in case of standard SVM, $Reg(w, b) = \frac{1}{2} \|w\|_2^2$). However, both $x_i$’s and $y_i$’s can be corrupted or uncertain. Corruption in $x_i$ can occur due to
noisy feature values or even missing feature values, while corruption in \( y_i \) is mainly due to incorrect label assignment. If uncertainty in \( x_i \) is only considered, SVM can be formulated as

\[
\begin{align*}
\min_{w, b, \xi} & \quad \text{Reg}(w, b) + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \quad y_i (w^\top x_i + b) \geq 1 - \xi_i, \quad \forall x_i \in \mathcal{U}_i, \quad i = 1, \ldots, n \\
& \quad \xi_i \geq 0, \quad i = 1, \ldots, n
\end{align*}
\]

(2.12)

Here uncertainty in \( x_i \) is described by uncertainty set \( \mathcal{U}_i \), i.e., all uncertain \( x_i \)s are coming from set \( \mathcal{U}_i \). This kind of optimization problem is known as uncertain optimization problem or (UOP) (2.1). The (UOP) is in fact a family of problems - one for each realization of \( x_i \). In the RO framework the family of problems, (UOP) is replaced by its robust counterpart (RC) (2.2) and the tractability of (RC) for this (UOP) depends on the properties of \( \mathcal{U}_i \).

**SVMs and Robust LP**  In [5], the authors considered \( L_1 \) regularization, i.e., \( \text{Reg}(w, b) = \|w\|_1 \) in (2.12) and polyhedral uncertainty i.e., for every \( x_i \), uncertainty in all feature values \( x_{ij} \) is bounded by an interval \( a_{ij} \leq x_{ij} \leq b_{ij} \) and derive constraints which ensure that all datapoints lying in the bounding hyper-rectangle of the \( i^{th} \) training datapoint (see Figure 2.3) are correctly classified. The bounding hyper-rectangle for \( i^{th} \) data point is denoted as :

\[
\mathcal{R}(a_i, b_i) \equiv \left\{ x = [x_1 \ldots x_d]^\top \mid a_{ij} \leq x_j \leq b_{ij} \right\}
\]

(2.13)

Here \( a_i \equiv [a_{i1} \ldots a_{id}]^\top \) and \( b_i \equiv [b_{i1} \ldots b_{id}]^\top \). The uncertain optimization problem is:

\[
\begin{align*}
\min_{w, b, \xi} & \quad \text{Reg}(w, b) + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \quad y_i (w^\top x_i + b) \geq 1 - \xi_i, \quad x_i \in \mathcal{R}(a_i, b_i), \quad i = 1, \ldots, d \\
& \quad \xi_i \geq 0, \quad i = 1, \ldots, n
\end{align*}
\]

(2.14)
The final tractable optimization problem equivalent to (2.14) solved by them was:

\[
\begin{align*}
\min_{w, b, \xi_i} & \quad \text{Reg}(w, b) + C\sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \quad y_i(w^\top m_i + b) \geq 1 - \xi_i + \|L_i w\|_1, \quad \xi_i \geq 0, \quad i = 1, \ldots, n
\end{align*}
\]  

(2.15)

where \( m_i \equiv \frac{b_i + a_i}{2} \) and \( L_i \) is a diagonal matrix with entries as \( L_{ij} \equiv \frac{b_{ij} - a_{ij}}{2} \). This is an example of Robust LP (A.2). The main disadvantage of this method is that it considers only support information of the uncertain data points and neglects any available statistical information, i.e., means or variance of uncertainty. While Chapter 3 and 4 of this Thesis tries to derive more generalized robust counterpart of SVM by considering both support and statistical information of the interval valued data.

![Figure 2.3: Robust Classifier for Hyper-Rectangular Uncertainty](image)

Figure 2.3: Robust Classifier for Hyper-Rectangular Uncertainty

On the other hand, in [27] the authors use robust optimization with polyhedral uncertainty set to address the problem where some of the features of the testing samples are missing. In contrast to all the previous works described here, it does not assume the presence of uncertainty
in the training example. Using a dummy feature to remove the bias term \( b \) and using the hinge loss, they have considered a nominal problem as shown below:

\[
\min_{\mathbf{w}} \text{Reg}(\mathbf{w}) + C \sum_{i=1}^{n} [1 - y_i \mathbf{w}^\top \mathbf{x}_i]_+
\]  

(2.16)

When at most \( M \) features are missing, in the worst case scenario \([1 - y_i \mathbf{w}^\top \mathbf{x}_i]_+\) can be replaced by:

\[
\max_{\mathbf{a}} [1 - y_i \mathbf{w}^\top (\mathbf{x}_i \ast (1 - \mathbf{a}_i))]_+
\]

s.t. \( a_{ij} \in \{0, 1\}, \quad \forall j = 1, \ldots, d, \)

\[\sum_{j=1}^{d} a_{ij} = M\]  

(2.17)

where \( \ast \) denotes pointwise vector multiplication. To avoid combinatorial part of this optimization problem, \( a_{ij} \in \{0, 1\} \) has been relaxed by the constraint \( 0 \leq a_{ij} \leq 1 \). Thus taking the dual of the maximization, and substituting into the original problem, they obtained the classifier that is maximally resistant up to \( M \) missing features:

\[
\min_{\mathbf{w}, \xi, \mathbf{z}, \mathbf{v}, t_i} \text{Reg}(\mathbf{w}) + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad y_i \mathbf{w}^\top \mathbf{x}_i - t_i \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n; \\
t_i \geq Mz_i + \sum_{j=1}^{d} v_{ij}, \quad v_i \geq 0, \quad i = 1, \ldots, n; \\
z_i + v_{ij} \geq y_ix_{ij}w_j, \quad i = 1, \ldots, n, j = 1, \ldots, d.
\]  

(2.18)

In [2], the authors consider \( L_1 \) regularization, i.e., \( \text{Reg}(\mathbf{w}, b) = \|\mathbf{w}\|_1 \) and the ellipsoidal uncertainty set (see Figure 2.4) given by:

\[\mathcal{U}_i = \{\mathbf{x}_i : (\mathbf{x}_i - \mu_i)^\top \Sigma_i (\mathbf{x}_i - \mu_i) \leq 1\}, \quad i = 1, \ldots, n;\]

where shape of the ellipsoid is determined by the covariance matrix \( \Sigma_i \) and center of it is determined by the mean \( \mu_i \). Therefore, each constraint becomes \( y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \forall \mathbf{x}_i \in \mathcal{U}_i\).
\( \mathcal{U}_i \) which is then approximated using Chebyshev inequality to \(^1\):

\[
y_i(w^\top \mu_i + b) \geq 1 - \xi_i \text{ and } y_i(w^\top \mu_i + b) \geq \|\Sigma_i^{\frac{1}{2}} w\|_2 - \xi_i
\]

Therefore, the final tractable robust formulation is:

\[
\begin{align*}
    \min_{w,b,\xi_i} & \quad \|w\|_1 + C \sum_{i=1}^n \xi_i \\
    \text{s.t.} & \quad y_i(w^\top \mu_i + b) \geq 1 - \xi_i, \\
                    & \quad y_i(w^\top \mu_i + b) \geq \|\Sigma_i^{\frac{1}{2}} w\|_2 - \xi_i, \\
                    & \quad \xi_i \geq 0, \ i = 1, \ldots, n
\end{align*}
\]

(2.19)

It is an instance of SOCP (for details please refer (A.3)). [28] solves the same problem

\(^1\Sigma_i = \Sigma_i^{\frac{1}{2}} \Sigma_i^{\frac{1}{2}}\)
with different regularization term $\text{Reg}(w, b) = \|w\|_2$. Although there are many SOCP solvers available now, they are based mainly on interior point method and hence need large amount of time and space to be executed successfully.

### 2.4.1 SVMs and Robust QP

In order to solve nonlinear SVM with uncertain data one has to solve the following Robust formulation:

$$
\begin{align*}
\max_{t_1, \alpha \in S_n} & \quad \alpha^T e - \frac{1}{2} t_1 \\
\text{s.t.} & \quad \alpha^T Y K Y \alpha \leq t_1, \quad \forall K \in \mathcal{U}_K
\end{align*}
$$

where $S_n = \{\alpha | 0 \leq \alpha_i \leq C, \sum_{i=1}^n \alpha_i y_i = 0\}$, $Y = \text{diag}(y_i)$ and the kernel matrix $K$ is uncertain and uncertainty in the kernel matrix is induced by the uncertainty in data points. (2.20) has a tractable robust counterpart only if the uncertain set $\mathcal{U}_K$ is ellipsoidal, i.e., $\mathcal{U}_K = \{K := \bar{K} + \sum_{l=1}^{L} u_l(K_l) : \|u\|_2 \leq 1\}$. Then the robust counterpart turns out to be a semi-definite program (SDP) (according to (A.5)):

$$
\begin{align*}
\max_{t_1, \alpha \in S_n} & \quad \alpha^T e - \frac{1}{2} t_1 \\
\text{s.t.} & \quad M \succeq 0
\end{align*}
$$

where

$$
M = \begin{pmatrix}
t_1 - \tau & \frac{1}{2} t_1 & \cdots & \frac{1}{2} t_1 & (K Y \alpha)^T \\
\frac{1}{2} t_1 & \tau & \cdots & \vdots & (K_1 Y \alpha)^T \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\frac{1}{2} t_1 & \cdots & \tau & (K_L Y \alpha)^T \\
K Y \alpha & K_1 Y \alpha & \cdots & K_L Y \alpha & I
\end{pmatrix}
$$

SDP has a very high time as well as space complexity and hence it is still difficult to solve Robust QP for large scale data. Chapter 7 has proposed efficient algorithms to solve Robust
SVM, which is robust to uncertainty in a kernel matrix.

2.4.2 SVMs and Chance Constraint Program

If the uncertainty in $x_i$ is stochastic, one can use robust formulation of SVM to find a classifier that satisfies the constraints with some probability (or chance). This is known as chance-constraint based approach. In [7], the authors considered two varieties of such chance constraints for $i = 1, \ldots, n$:

(a) $\text{Prob}_{x_i \sim \mathcal{N}(\mu, \Sigma)}(y_i(w^\top x_i + b) \geq 1 - \xi_i) \geq 1 - \varepsilon_i$

(b) $\inf_{x_i \sim (\mu, \Sigma)} \text{Prob}_{x_i}(y_i(w^\top x_i + b) \geq 1 - \xi_i) \geq 1 - \varepsilon_i$ (2.23)

Constraint (2.23(a)) controls the probability of constraint violation, when the uncertainty follows a known Gaussian distribution. Constraint (2.23(b)) is more conservative and it controls the worst-case probability of constraint violation, over all distributions with mean $\mu_i$ and variance $\Sigma_i$. For both the cases, robust constraint can be approximated by:

$$y_i(w^\top \mu_i + b) \geq 1 - \xi_i - \kappa_i \|\Sigma_i^{\frac{1}{2}} w\|.$$  

Where in the Gaussian case (2.23(a)) $\kappa_i = \phi^{-1}(\varepsilon_i)$, $\phi$ is the standard Gaussian c.d.f. and for (2.23(b)) $\kappa_i = \sqrt{\frac{\varepsilon_i}{1-\varepsilon_i}}$. [7] has also given a geometric interpretation of this robust formulation along with its robustness to the estimation error for parameters. Again [29, 7] used (2.23(a)) to address the missing value problem. Since their optimization setting allows uncertainty in terms of a Normal distribution they assumed that $(x|y)$ is jointly Normal with mean $\mu_y$ and covariance $\Sigma_y$. By means of EM algorithm they first estimated $\mu_y$ and $\Sigma_y$ from incomplete data and then they used the conditionally Normal estimates of $(x_{\text{missing}}|x_{\text{observed}}, y)$ in the optimization problem.

But use of Chebyshev Inequality results into more conservative robust classifiers with less margin. This Thesis is going to explore the application of a novel Bernstein like inequality to build non-overly conservative robust classifier with wider classification margin.
Chapter 3

Robust Classifiers for Interval-Valued Data under Support Information: A Chance Constraint Approach

Abstract

This chapter presents a novel methodology for constructing maximum-margin classifiers which are robust to interval-valued uncertainty in data. The idea is to employ chance-constraints which ensure that the uncertain data points are classified correctly with high probability. The key novelty is in employing Bernstein bounding schemes to relax the resulting chance-constrained program as a convex second order cone program. The Bernstein based relaxations presented in the chapter require only the knowledge of support and mean of the uncertain examples alone and make no assumptions on distributions regarding the underlying uncertainty. Classifiers built using the proposed methodology model interval-valued uncertainty in a less conservative fashion and hence are expected to generalize better than existing methods. Experimental results on synthetic and real-world datasets show that the proposed classifiers are better equipped to handle interval-valued uncertainty than state-of-the-art.

3.1 Introduction

In the recent past there has been a growing interest in analysis of interval-valued data in the learning community [3, 32]. As discussed previously in Section 1.2, in many real world problems it is not possible to describe the data by a precise value but intervals may be a more
Chapter 3. Robust Classifiers for Interval-Valued Data

proper description. For example, cancer diagnosis data\footnote{Examples are the Wisconsin breast cancer diagnostic/prognostic datasets available at http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)} and micro-array data. Past research has shown that handling uncertainty in such applications by the representation as interval data leads to accurate learning algorithms [5, 3]. Classification formulations which are capable of handling interval data have immense importance from a pragmatic perspective. This chapter presents a maximum-margin classification formulation which uses means and bounding hyper-rectangles (support) of the interval-valued training examples in order to build the linear decision function. As shown in this Chapter, the proposed linear classifier is robust to interval uncertainty and is also not overly-conservative.

3.1.1 Review of Past Work

This section presents a review of the work done on maximum-margin classification with uncertain data\footnote{Nomenclature of formulations: prefix “MM” denotes Maximum Margin classifier and “IC” denotes Interval-valued Classifier. Partial information of Support, Mean employed by the classifier are denoted by ‘S’, ‘M’ respectively.}. The discussion is started by the well known SVM formulation [1], which assumes that the training datapoints are known exactly. Here, a hyperplane, $\mathbf{w}^T \mathbf{x} + b = 0$, that maximally separates the positive and negative training datapoints is constructed. Denoting the training datapoints by $\mathbf{x}_i = [x_{i1} \ldots x_{id}]^T$, $i = 1, \ldots, n$ and their class labels by $y_i$, this problem can be expressed as:

$$\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{n} \xi_i$$

subject to $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$, $\xi_i \geq 0$, $i = 1, \ldots, n$ (3.1)
However, if the training datapoints, $x_i$'s, are uncertain, the SVMs fail to generalize well [7, 5]. The simplest way to handle uncertainty while working under the maximum-margin principle is by employing the means, $\mu_i \equiv E[x_i]$, for building the optimal hyperplane. This can be written as:

\[
\text{(MM-M)} \quad \min_{w,b,\xi_i} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad y_i(w^T \mu_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n \quad (3.2)
\]

Assuming uncertainty in the datapoints is bounded i.e., the support is known, tractable classification formulations which are robust to uncertainty can been derived [5, 6]. Specifically, [5] assume that $a_{ij} \leq x_{ij} \leq b_{ij}$ and derive constraints which ensure that all datapoints lying in the bounding hyper-rectangle of the $i$th training datapoint:

\[
R(a_i, b_i) = \left\{ x = [x_1 \ldots x_d]^T \mid a_{ij} \leq x_j \leq b_{ij} \right\}
\]

are classified correctly. Here $a_i \equiv [a_{i1} \ldots a_{id}]^T$ and $b_i \equiv [b_{i1} \ldots b_{id}]^T$. This leads to the following maximum-margin formulation:

\[
\text{(MM-S)} \quad \min_{w,b,\xi_i} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad y_i(w^T m_i + b) \geq 1 - \xi_i + \|L_i w\|_1, \quad \xi_i \geq 0, \quad i = 1, \ldots, n \quad (3.4)
\]

where $m_i \equiv \frac{b_i + a_i}{2}$ and $L_i$ is a diagonal matrix with entries as $l_{ij} = \frac{b_{ij} - a_{ij}}{2}$.

Each of the two robust formulations presented above differ in the way uncertainty is modeled using various partial information like support and first order moments. The formulation (MM-M) uses only mean (first order moment) information, while (MM-S) uses support information. The conservative nature of a formulation depends on the partial information employed by it. When more information is employed, then the uncertainty can be modeled in a better way— leading to robust as well as non-overly-conservative classifiers. Now, the conservative nature of a robust classifiers has direct influence over the generalization ability of the classifier — this is justified in the following text. Note that, more conservative the uncertainty modeling
is, less tighter the classification constraints are in the respective formulations. For example, (MM-S) models the uncertain datapoint using its bounding hyper-rectangle whereas (MM-M) models it as the single point \( \mu_i \). Clearly, the classification constraints (3.4) in (MM-S) which imply that the entire hyper-rectangle, that must be classified correctly, are more conservative than those in (MM-M) which imply that the mean, \( \mu_i \), alone needs to be classified correctly. It is also easy to see that because of this conservative modeling of uncertainty in (MM-S), the margin, \( \frac{1}{2} ||w||_2^2 \), achieved by it is less than that with (MM-M). According to the structural risk minimization principle [1], larger the margin of a classifier is, better is its generalization ability. Thus (MM-S) in-spite-of being robust to uncertainty fails to generalize well due to its conservative nature. On the other hand, (MM-M), though models uncertainty in a less conservative manner, it is not robust enough as it assumes mean is the only possible position for the uncertain datapoint. Thus in order to achieve good generalization, classifiers need to be robust to uncertainties in data while not being overly-conservative.

### 3.1.2 Contribution

The idea is to model interval-valued uncertainty using Chance-Constrained Programming (CCP). The main contribution of this chapter is to approximate the CCP as a Second Order Cone Program (SOCP) using Bernstein schemes [10]. To the best of our knowledge, this is the first time Bernstein approximation schemes are employed for relaxing CCP based learning formulations. SOCPs are well studied convex optimization problems with efficient interior point solvers [12] (e.g. SeDuMi [33]). The key advantage of the Bernstein scheme is that, no assumptions on distributions regarding the underlying uncertainty are made and only partial information like support and mean of the uncertain examples is required. Geometric interpretation of the SOCP formulation reveals that the classifier views each example with interval uncertainty as a region of intersection of its bounding hyper-rectangle and an ellipsoid centered at its mean. Thus the proposed classifier is far less conservative than the methods which utilize the bounding hyper-rectangle information alone. Since a classifier’s conservativeness directly affects the classification margin achieved, the proposed classifier is expected to generalize better. Methodology of classifying uncertain test examples is discussed and error measures for evaluating performance
of interval data classifiers are also presented. Experimental results show that the proposed classifier outperforms state-of-the-art when evaluated using any of the discussed error measures.

3.1.3 Structure of the Chapter

This chapter is organized as follows: in Section 3.2, the main contributions of the chapter are presented. In Section 3.4, experiments on real-world and synthetic data are presented. Section 3.5 summarizes the work and concludes the chapter.

3.2 Robust Classifiers for Interval-Valued Uncertainty

This section presents the main contribution of this chapter, a novel maximum-margin formulation for interval data in Section 3.2.1. A discussion on geometric interpretation of the proposed formulation is presented in Section 3.2.2. The section concludes with a discussion on error measures which evaluate the performance of a classifier on interval data.

3.2.1 Maximum-Margin Formulation using Bernstein Bounds

In this section, a maximum-margin classification formulation for interval data, which assumes the mean and the bounding hyper-rectangles are known for each example, is presented. It is also assumed that the features describing the data are independent. Let \( x_i = [x_{i1} \ldots x_{id}]^\top \) be the random vector representing \( i \)-th training example (\( d \) denotes dimensionality of the data) and \( y_i \) denotes its label (\( i = 1, \ldots, n \)). Let \( a_i = [a_{i1} \ldots a_{id}]^\top \), \( b_i = [b_{i1} \ldots b_{id}]^\top \) and \( a_{ij} \leq x_{ij} \leq b_{ij} \), so that \([a_i, b_i]\) represents the bounding hyper-rectangle of \( i \)-th example. Also let \( E[x] \) denotes mean of the random variable \( x \). Also let \( l_{ij} = \frac{b_{ij} - a_{ij}}{2} \) and the diagonal matrix containing entries as \( l_{ij} \), \( j = 1, \ldots, d \) be denoted by \( L_i \). Denote the mid-points of the bounding intervals of \( x_{ij} \) by \( m_{ij} = \frac{a_{ij} + b_{ij}}{2} \) and the vector with entries \( m_{ij}, j = 1, \ldots, d \) as \( m_i \). Assume that the means, \( E[x_i] \), are denoted by \( \mu_i = [\mu_{i1} \ldots \mu_{id}]^\top \). Let \( \hat{\mu}_{ij} = \frac{\mu_{ij} - m_{ij}}{l_{ij}} \) and assume \( \hat{\mu}_i \) denotes the vector with entries \( \hat{\mu}_{ij}, j = 1, \ldots, d \).
Consider the problem of constructing a maximum-margin linear classifier using the training example $x_i$, which have interval-valued uncertainty. Let the discriminating hyperplane be denoted by $w^T x + b = 0$. Then the constraints $y_i(w^T x_i + b) \geq 1$ ensure that the training data is classified correctly. Slack variables, $\xi_i \geq 0$, can be introduced in order to handle outliers. Thus the classification constraints turn out to be $y_i(w^T x_i + b) \geq 1 - \xi_i$. Since the constraints involve the random vector, $x_i$, they cannot be satisfied always. Hence, alternatively, one can ensure that the following chance-constraints are satisfied:

$$\text{Prob} \left( y_i(w^T x_i + b) \leq 1 - \xi_i \right) \leq \varepsilon$$

where $0 \leq \varepsilon \leq 1$ is a small number denoting the upper bound on misclassification error made on an example and is a user-given parameter. Using these constraints the following maximum-margin formulation, similar in spirit to SVMs [1], can be written:

$$(\text{CCP}) \quad \min_{w, b, \xi_i} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

s.t. $\text{Prob} \left( y_i(w^T x_i + b) \leq 1 - \xi_i \right) \leq \varepsilon$, $\xi_i \geq 0$, $i = 1, \ldots, n$  \hspace{1cm} (3.6)

The following section will show that the (CCP) (3.6) can be approximated as an SOCP problem by using Bernstein bounds. To this end, the following theorem is presented, which specializes the Bernstein approximation schemes described in [10, 11]:

**Theorem 3.1.** Consider the following notation ($\forall i = 1, \ldots, n$, $j = 1, \ldots, d$):

$$l_{ij} = \frac{b_{ij} - a_{ij}}{2} \quad m_{ij} = \frac{b_{ij} + a_{ij}}{2} \quad \mu_{ij} = E[x_{ij}]$$

$$L_i = \text{diag} ([l_{i1} \ldots l_{id}]) \quad m_i = [m_{i1} \ldots m_{id}]^T \quad \mu_i = [\mu_{i1} \ldots \mu_{id}]^T$$

$$\hat{\mu}_{ij} = \frac{E[x_{ij}] - m_{ij}}{l_{ij}} \quad \hat{\mu}_i = [\hat{\mu}_{i1} \ldots \hat{\mu}_{id}]^T \quad \hat{\Sigma}_i = \text{diag} ([\nu(\hat{\mu}_{i1}) \ldots \nu(\hat{\mu}_{id})])$$

where $\nu(\hat{\mu}_{ij})$ is given by:

$$\nu(\hat{\mu}_{ij}) = \min \left\{ \nu \geq 0 \left| \frac{\nu^2}{2} r^2 + \hat{\mu}_{ij} t - \log \left( \cosh(t) + \hat{\mu}_{ij} \sinh(t) \right) \geq 0, \forall t \in \mathbb{R} \right. \right\}$$

(3.8)
The chance constraint (3.5), which represents the classification constraint for \( i \)th example, is satisfied if the following cone constraint, in variables \( w, b, \xi_i, z_i \), holds:

\[
y_i(w^T \mu_i + b) + z_i \mu_i \geq 1 - \xi_i + \|z_i\|_1 + \sqrt{2 \log(1/e)} \|\Sigma_i (y_i L_i w + z_i)\|_2
\]  

(3.9)

Proof. The chance-constraint (3.5) can be written as:

\[
\text{Prob} \left( -y_i w^T x_i + (1 - \xi_i + y_i b) \geq 0 \right) \leq \epsilon
\]

Now, let variables \( u_i, v_i \) be chosen such that:

\[
u_i + v_i = -y_i w
\]

(3.10)

Since \( a_i \leq x_i \leq b_i \), and hence \( v_i^T x_i \leq v_i^T m_i + \|L_i v_i\|_1 \) holds. Using this inequality, the chance-constraint (3.5) is satisfied if:

\[
\text{Prob} \left( u_i^T x_i + u_{i0} \geq 0 \right) \leq \epsilon
\]

(3.11)

where \( u_{i0} = 1 - \xi_i + y_i b + v_i^T m_i + \|L_i v_i\|_1 \). Clearly, the advantage of introducing the variables \( u_i, v_i \) (3.10) is to utilize the bounding hyper-rectangle information via the inequality \( v_i^T x_i \leq v_i^T m_i + \|L_i v_i\|_1 \) (also see Lemma 3.2).

Using Markov inequality and independence of random variables \( x_{ij}, j = 1, \ldots, d \) gives:

\[
\text{Prob} \left( u_i^T x_i + u_{i0} \geq 0 \right) \leq \exp\{\zeta u_{i0}\} \prod_j \mathbb{E}[\exp\{\zeta u_{ij}\}], \forall \zeta \geq 0
\]

(3.12)

The Key of modeling chance constraint (3.11) now depends on how one upperbounds the moment generating functions \( \mathbb{E}[\exp\{t x_{ij}\}], t \in \mathbb{R} \). To this end, the following lemma is used:

Lemma 3.1. Consider the notation in (3.7). Then,

\[
\mathbb{E}[\exp\{t x_{ij}\}] \leq \exp \left\{ \frac{v(\mu_{ij})^2 l_{ij}^2}{2} t^2 + \mu_{ij} t \right\} \forall t \in \mathbb{R}
\]

(3.13)
Proof. The fact that \( \exp\{tx_{ij}\} \) is a convex function gives the following inequality: \( \exp\{tx_{ij}\} \leq \frac{b_{ij} - x_{ij}}{b_{ij} - a_{ij}} \exp\{ta_{ij}\} + \frac{x_{ij} - a_{ij}}{b_{ij} - a_{ij}} \exp\{tb_{ij}\} \). Taking expectation on both sides and re-writing the resulting inequality in terms of \( m_{ij}, l_{ij} \) gives:

\[
E[\exp\{tx_{ij}\}] \leq \exp\{m_{ij}t + h_{ij}(l_{ij}t)\} \tag{3.14}
\]

where \( h_{ij}(\tilde{t}) \equiv \log \left( \cosh(\tilde{t}) + \hat{\mu}_{ij} \sinh(\tilde{t}) \right) \). Note that, \( h_{ij}(0) = 0, h_{ij}'(0) = \hat{\mu}_{ij} \) and \( h_{ij}''(\tilde{t}) \leq 1, \forall \tilde{t} \). This gives the inequality \( h_{ij}(\tilde{t}) \leq \frac{1}{2} \tilde{t}^2 + \hat{\mu}_{ij} \tilde{t}, \forall \tilde{t} \). In fact, using \( v(\hat{\mu}_{ij}) \) as defined in (3.8), the tighter inequality can be obtained, \( h_{ij}(\tilde{t}) \leq \frac{v(\hat{\mu}_{ij})^2}{2} \tilde{t}^2 + \hat{\mu}_{ij} \tilde{t}, \forall \tilde{t} \). Using this inequality in (3.14), and noting that \( \mu_{ij} = l_{ij}\hat{\mu}_{ij} + m_{ij} \), (3.13) is obtained. This completes the proof of Lemma 3.1.

Using Lemma 3.1, from (3.12) one can get:

\[
\log \left[ Prob \left( u_i^\top x_i + u_{i0} \geq 0 \right) \right] \leq \xi(u_{i0} + u_i^\top \mu_i) + \frac{\xi^2}{2} \| L_i \hat{\Sigma}_i u_i \|_2^2, \forall \xi \geq 0
\]

Since this inequality holds for all values of \( \xi \), if one can ensure that for certain \( \xi \) the right-hand side of the inequality is \( \leq \log(\epsilon) \), then it would satisfy the chance-constraint (3.11).

Choosing \( \xi^* = -\frac{m_{i0} + u_i^\top \mu_i}{\| L_i \hat{\Sigma}_i u_i \|_2^2} \), which is the one minimizing right-hand side of the above inequality,(3.11) and in turn (3.5) are satisfied if:

\[
u_{i0} + u_i^\top \mu_i + \sqrt{2\log(1/\epsilon)}\| L_i \hat{\Sigma}_i u_i \|_2 \leq 0 \tag{3.15}
\]

Substituting the value of \( u_{i0} \), eliminating the variable \( u_i \) from (3.10), (3.15) and introducing the variable \( z_i = L_i v_i \) gives (3.9). This completes the proof of the theorem.

The values of \( v(\hat{\mu}_{ij}) \) (3.8) can be calculated numerically (refer Appendix B.1). Using Theorem 3.1 and CCP (3.6), a maximum-margin SOCP formulation for interval data which ensures that the probability of misclassification is less than \( \epsilon \), can be written as follows:
\[
(\text{IC-SM}) \quad \min_{w,b,z_i} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad y_i(w^T \mu_i + b) + z_i^T \hat{\mu}_i \geq 1 - \xi_i + \|z_i\|_1 + \kappa \|\Sigma_i (y_i L_i w + z_i)\|_2, \xi_i \geq 0
\]

(3.16)

where \( \kappa = \sqrt{2 \log(1/\varepsilon)} \) and \( \hat{\mu}_i, \mu_i, L_i, \hat{\Sigma}_i \) are as given in (3.7). As mentioned earlier, \( C \) and \( \varepsilon \) are user-given parameters. Problem (3.16) can be efficiently solved using generic cone solvers like SeDuMi\(^3\), Mosek\(^4\) or CPLEX\(^5\).

Assuming that the uncertain data is represented using replicates for each training example (as in case of micro-array data), following are the steps involved in training with the proposed classification formulation (3.16):

- For each training example, estimate \( l_{ij}, m_{ij}, \mu_{ij} \) and \( v(\hat{\mu}_{ij}) \) (3.7-3.8). This can be done using the replicates given for each example.

- Solve (3.16) for optimal values of \( (w, b) \) using any SOCP solver like SeDuMi.

### 3.2.2 Geometric Interpretation of the Formulation

In this section, a geometrical interpretation for the proposed formulation (3.16) is presented. To this end, consider the following lemma:

**Lemma 3.2.** Consider the notation in (3.7) and let \( \kappa = \sqrt{2 \log(1/\varepsilon)} \). Suppose \( \mathcal{E}(\mu_i, \kappa L_i \hat{\Sigma}_i) \) represents the ellipsoid \( \{ x = \mu_i + \kappa L_i \hat{\Sigma}_i u : \|u\|_2 \leq 1 \} \) and \( \mathcal{R}(a_i, b_i) \) represents the hyper-rectangle \( \{ x : a_i \leq x \leq b_i \} \). Consider the problem of correctly classifying points belonging

---

\(^3\)Available at [http://sedumi.mcmaster.ca/](http://sedumi.mcmaster.ca/)


3.17 The continuum of constraints represented in (3.17) is satisfied if and only if the constraint (3.9) holds.

Proof. The constraint (3.17) hold if and only if:

\[ 1 - \xi_i + y_i b + \left( \max_{x \in E(\mu_i, kL_i \hat{\Sigma}_i) \cap R(a_i, b_i)} -y_i w^T x \right) \leq 0 \]

Note that, \( \max_{x \in E(\mu_i, kL_i \hat{\Sigma}_i) \cap R(a_i, b_i)} (-y_i w^T x) \) is the support function of the set \( E(\mu_i, kL_i \hat{\Sigma}_i) \cap R(a_i, b_i) \) (denoted by \( I_{E(\mu_i, kL_i \hat{\Sigma}_i) \cap R(a_i, b_i)} (-y_i w) \)). Since support function of intersection of two sets is the infimal convolution of support functions of the individual sets (see Section 16, [34]), one can have that,

\[ I_{E(\mu_i, kL_i \hat{\Sigma}_i) \cap R(a_i, b_i)} (-y_i w) = \inf \left\{ I_{E(\mu_i, kL_i \hat{\Sigma}_i)} (u_i) + I_{R(a_i, b_i)} (v_i) \mid u_i + v_i = -y_i w \right\} \]

Thus:

\[ (3.17) \iff 1 - \xi_i + y_i b + \inf \left\{ I_{E(\mu_i, kL_i \hat{\Sigma}_i)} (u_i) + I_{R(a_i, b_i)} (v_i) \mid u_i + v_i = -y_i w \right\} \leq 0 \]

\[ \iff \exists \ u_i, v_i \ni u_i + v_i = -y_i w, \]

\[ 1 - \xi_i + y_i b + \left\{ I_{E(\mu_i, kL_i \hat{\Sigma}_i)} (u_i) + I_{R(a_i, b_i)} (v_i) \right\} \leq 0 \]  

(3.18)

Now it is easy to see that \( I_{E(\mu_i, kL_i \hat{\Sigma}_i)} (u_i) = u_i^T \mu_i + k \|L_i \hat{\Sigma}_i u_i\|_2 \) and \( I_{R(a_i, b_i)} (v_i) = v_i^T m_i + \|L_i v_i\|_1 \). Substituting these values in (3.18) and noting that \( \mu_i = L_i \hat{\mu}_i + m_i \), gives (3.10, 3.15).

Since the constraints (3.10, 3.15) are equivalent to (3.9) (see proof of Theorem 3.1), then (3.17) \( \iff \) (3.9). This completes the proof. \[ \square \]

The above lemma shows that the proposed classifier (3.16) views each interval data example as the intersection of the bounding hyper-rectangle and an ellipsoid centered at its mean.
Chapter 3. Robust Classifiers for Interval-Valued Data

Figure 3.1: Geometric Interpretation of the Formulation

with semi-axis lengths proportional to \( l_{ij} \sqrt{\hat{v}(\hat{\mu}_{ij})} \). A pictorial presentation of geometric interpretation is available in Figure 3.1. The proposed classifier considers each interval datapoint as the shaded region around the data point, which is the intersection of bounding hyper-rectangle and an ellipsoid. In this way the proposed formulation takes into account both the mean and bounding hyper-rectangle informations. Note that, Lemma 3.2 theoretically proves that the proposed classifier is always less conservative (pessimistic) than classifiers which use the bounding hyper-rectangle information alone [5] (this is because \( \mathcal{E}(\mu, kL, \Sigma) \cap \mathcal{R}(a, b) \subset \mathcal{R}(a, b) \)). In fact, it is easy to see that classifiers which utilize the bounding hyper-rectangle information alone and classifiers which utilize the mean information alone are special cases of the proposed formulation (with \( \epsilon = 0 \) and \( \epsilon = 1 \) respectively).

It is interesting to note the effect of \( v(\hat{\mu}_{ij}) \) (3.8) on the proposed classifier. As mentioned earlier, the semi-axis lengths of the uncertainty ellipsoid are proportional to \( v(\hat{\mu}_{ij}) \). Table B.1 and Figure 3.2 shows that as \( \hat{\mu} \) increases from 0 to 1, \( v(\hat{\mu}) \) decreases from 1 to 0. In other words, as the mean of example shifts from center to a corner of the bounding hyper-rectangle, the size of the uncertainty ellipsoid decreases. This is very intuitive because, in one extreme case where mean lies at a corner, the datapoint is deterministic and in the other extreme case, where mean is at center of the hyper-rectangle, the uncertainty of the datapoint is maximum. This phenomenon is also illustrated in Figure 3.3, where the bounding hyper-rectangle and
the uncertainty region at various positions of the mean are shown. It can be seen that, as the mean moves towards a corner, not only the uncertainty region moves but also its size decreases. However, a classifier which does not employ the mean information [5], always views the example as the whole hyper-rectangle. Thus the proposed classifier is robust to interval-valued uncertainty, as well as not overly-conservative.

### 3.3 Prediction rules and Error metrics

This section presents the methodology for labeling interval-valued test examples and discusses error measures for evaluating the performance of interval data classifiers. Depending on the form in which the examples are available, the labeling methodologies summarized in Table 3.1 can be employed. Here, $y_i^{pr}$ denotes the predicted label for test example $x_i$ (also refer (3.7) for notation). Please note that, all the methodologies summarized in Table 3.1 actually try to consider the predicted label of the majority portion of the uncertain region as predicted label for test example. Once a test example is labeled using the appropriate methodology, the overall misclassification error for the given test dataset can be calculated as the percentage of examples in which $y_i^{pr}$ and $y_i$ do not agree:

$$\text{MajErr} = \frac{\sum \mathbb{1}_{y_i^{pr} \neq y_i}}{\# \text{test examples}} \times 100$$  \hspace{1cm} (3.19)
Figure 3.3: Figure showing bounding hyper-rectangle and uncertainty sets for different positions of mean. Mean and boundary of uncertainty set marked with same color.

Table 3.1: Table summarizing ways of representing interval-valued uncertainty in examples and corresponding label prediction methodologies.

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Form of Uncertainty Information</th>
<th>Labeling Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\mathcal{R}(a_i, b_i)$ are given</td>
<td>$y_i^{pr} \equiv \text{sign}(w^T m_i + b)$</td>
</tr>
<tr>
<td>2</td>
<td>$\mathcal{R}(a_i, b_i), \mu_i$ are given</td>
<td>$y_i''' \equiv \text{sign}(w^T \mu_i + b)$</td>
</tr>
<tr>
<td>3</td>
<td>Set of replicates $r_{ij}, r_{i2}, \ldots$</td>
<td>$y_i''''$ is label of majority of $r_{ij}$. Label of $r_{ij}$ is $\text{sign}(w^T r_{ij} + b)$</td>
</tr>
</tbody>
</table>

Note that, the proposed formulation can be solved when the training examples are in any of the 3 forms shown in Table 3.1. In case of form 2, the support and mean information are readily available and in case of form 3, these partial information can be easily estimated from the replicates. In case of form 1, since no mean information is available the proposed formulation can be solved using $\epsilon = 0$, which as discussed in Section 3.2.2 is the maximum-margin classifier built using support information alone.

Based on the discussion in Section 3.2.2, another interesting error measure can be derived. Given an uncertain test example $x_i$ with label $y_i$, one can calculate the value of $\epsilon = \epsilon_{opt} = \exp \left\{ -\frac{(w^T \mu_i + b)^2}{2(w^T L_i \Sigma_i w)} \right\}$ for which the uncertainty ellipsoid $\mathcal{E}((\mu_i, \kappa L_i \Sigma_i))$ touches the discriminating hyperplane, $w^T x + b = 0$. Additionally, if true label, $y_i$, of the example is same as the
predicted label \( (y_i^{pr}) \), then the proof of Theorem 3.1 shows that the true probability of misclassification of the test example is less than or equal to \( \varepsilon_{opt} \). This leads to the following definition of error on each test example:

\[
\text{OptErr}_i = \begin{cases} 
1 & \text{if } y_i \neq y_i^{pr} \\
\varepsilon_{opt} & \text{if } y_i = y_i^{pr} \text{ and } \exists \mathbf{x} \in \mathcal{R}(a_i, b_i) \ni y_i (\mathbf{w}^T \mathbf{x} + b) < 0 \\
0 & \text{if } y_i (\mathbf{w}^T \mathbf{x} + b) \geq 0 \forall \mathbf{x} \in \mathcal{R}(a_i, b_i)
\end{cases}
\] (3.20)

The overall error, \( \text{OptErr} \), can be calculated as percentage of \( \text{OptErr}_i \), over all test examples:

\[
\text{OptErr} = \frac{\sum_i \text{OptErr}_i}{\# \text{test examples}} \times 100
\] (3.21)

Note that, both \( \text{MajErr} \) and \( \text{OptErr} \) can be estimated for any hyperplane classifier and are not specific to the proposed classifier. Experimental results show that the proposed classifier achieves lower \( \text{MajErr} \) and \( \text{OptErr} \) when compared to existing interval data classification methods.

### 3.4 Experimental Evaluation

In this section, experimental results which compare performance of the proposed interval data classifier (3.16) (denoted by \( \text{IC-SM} \)) and the maximum-margin classifier which utilizes bounding hyper-rectangle information alone [5] (denoted by \( \text{MM-S} \)) as defined in (3.4). In particular, it would be interesting to explore the following questions.

1. Comparison of the margins \( 2/\|\mathbf{w}\|_2 \) achieved by \( \text{MM-S} \) and \( \text{IC-SM} \)
2. Comparison of performance of \( \text{IC-SM} \) against \( \text{MM-M} \), \( \text{MM-R} \) and \( \text{MM-S} \)

Note that, the only difference between (3.4) and the one proposed in [5] is minimization of \( \|\mathbf{w}\|_2 \) in the objective rather than minimization of \( \|\mathbf{w}\|_1 \), which implies maximum-margin classification rather than sparse classification. This was done in order to achieve a fair comparison of the methodologies. Traditional classifiers like SVMs cannot handle interval-valued data.
Chapter 3. Robust Classifiers for Interval-Valued Data

However, when the means of the uncertain examples are known or uncertainty is represented using replicates (e.g. form 2 and 3 in Table 3.1 respectively), SVMs can be trained by considering mean of each example as a training datapoint (denoted by MM-M (3.2)) or by considering each replicate as a training datapoint (denoted by MM-R). Hence, wherever applicable, comparison of the performance of the proposed classifier with SVM based methods is also done.

Section 3.4.2 compares classification margins \(\frac{2}{\|w\|_2}\) achieved by MM-S and IC-SM, which represents state-of-the-art and the proposed interval data classifiers respectively. The key results of this chapter are presented in Section 3.4.3. These results compare the MajErr (3.19) and OptErr (3.21) obtained with various classifiers.

### 3.4.1 Datasets

Experiments were performed on synthetic datasets and two real-world datasets:

**WBCD** Wisconsin Breast Cancer Diagnostic dataset\(^6\): The task is to classify “benign” and “malignant” tumours based on 10 features computed from tumour cell nuclei. However, since the measurements are not the same over all tumour cells, the mean, standard-error and maximum values of the 10 features are provided. From this information the support and mean for each training datapoint are estimated.

**Micro-array**: Task is to identify four kinds of drugs: Azoles (\(A\)), Fibrates (\(F\)), Statins (\(S\)) and Toxicants (\(T\)) based on gene-expression data\(^7\) [3]. Since the experiments are noisy, three replicates of each datapoint are provided. Instead of handling a multi-class problem, six binary classification tasks are defined using “one versus one” scheme (e.g. \(A\) vs. \(F\) and so on). As a preprocessing step, the dimension of the problem is reduced to 166 by feature selection using Fisher score.

---

\(^6\)Available at [http://mlearn.ics.uci.edu/MLSummary.html](http://mlearn.ics.uci.edu/MLSummary.html)

Synthetic: Generation methodology: a) nominal (true) datapoints were generated using Gaussian mixture models b) uncertainty was introduced into each nominal point using standard finite-supported distributions (whose parameters were chosen randomly) c) replicates for each nominal datapoint were produced by sampling the chosen noise distribution. The synthetic datasets are named using dimension of the dataset and are subscripted with the distribution used for generating replicates (e.g. synthetic data of dimensionality \(d\) with Uniform, and truncated Beta noise distributions with \(N\) data point in each class and \(L\) samples for each data point are denoted by \(D_{dU}(S,N,L)\) and \(D_{dB}(S,N,L)\), where \(S\) denotes number of dataset with same distribution and dimensionality were created to calculate average performance. In each case, synthetic test data was also generated independently.

Platform and package: The proposed formulation IC-SM and MM-S have been implemented in Matlab with the help of Sedumi\(^8\) \cite{sedumi} and standard QP solver quadprog \(^9\) respectively. libSVM \(^10\) has been used as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 4 Intel Core(TM) 2 Quad 2.40 GHz processors and 4GB of RAM.

3.4.2 Comparison of Margin

In this section, the margins (\(2/\|w\|_2\)) achieved by MM-S and IC-SM at a fixed value of the \(C\) parameter are compared.

Datasets: Dataset were used here are \(D_{2U}(5,100,100)\), \(D_{10U}(5,100,100)\), \(D_{2B}(5,100,100)\) and \(D_{10B}(5,100,100)\).

Results: Figure 3.4 summarizes the results. Note that, at all values of \(\epsilon\), the classification margin with IC-SM is higher than that with MM-S. Also, as the value of \(\epsilon\) or dimensionality of the data increases, difference in the margins achieved by MM-S and IC-SM also increases.

\(^8\)http://sedumi.ie.lehigh.edu/
\(^10\)http://www.csie.ntu.edu.tw/~cjlin/libsvm/
The explanation for this is clear from the geometric interpretation presented in Section 3.2.2. According to structural risk minimization principle of Vapnik [1], higher margin implies better generalization. Hence the proposed classifier is expected to achieve good generalization for interval data. As a baseline for comparison, the margin achieved by the SVM trained using means of the examples, MM-M, is also shown in the figure. Since MM-M does not take into account the interval uncertainty and assumes the mean to be the “true” training example, it always achieves higher margin than MM-S, IC-SM. The trend shown in Figure 3.4 remains the same for higher dimensions and with different noise distributions (\(D_d B\)).

Figure 3.4: Figure comparing margins achieved by MM-M, MM-S and IC-SM at various \(\varepsilon\) values (\(D_2 U(5, 100, 100)\) on top left, \(D_{10} U(5, 100, 100)\) on top-right, \(D_2 B(5, 100, 100)\) on bottom-left and \(D_{10} B(5, 100, 100)\) on bottom-right, ).
Chapter 3. Robust Classifiers for Interval-Valued Data

Table 3.2: Table comparing classification error ($100\times ErrorMeasure\%$) $MajErr$ and $OptErr$ obtained with MM-M, MM-R, MM-S and IC-SM.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_{10U}$</td>
<td>32.07</td>
<td>59.90</td>
<td>44.80</td>
<td>65.70</td>
<td>51.05</td>
<td>53.62</td>
<td>20.36</td>
<td>52.68</td>
</tr>
<tr>
<td>$D_{10B}$</td>
<td>46.46</td>
<td>54.78</td>
<td>48.02</td>
<td>53.52</td>
<td>46.67</td>
<td>49.50</td>
<td>46.18</td>
<td>49.38</td>
</tr>
<tr>
<td>$A$ vs. $T$</td>
<td>00.75</td>
<td>46.47</td>
<td>00.08</td>
<td>46.41</td>
<td>55.29</td>
<td>58.14</td>
<td>00.07</td>
<td>39.68</td>
</tr>
<tr>
<td>$A$ vs. $F$</td>
<td>09.02</td>
<td>64.64</td>
<td>08.65</td>
<td>68.56</td>
<td>61.69</td>
<td>61.69</td>
<td>06.10</td>
<td>39.63</td>
</tr>
<tr>
<td>$A$ vs. $T$</td>
<td>12.92</td>
<td>73.88</td>
<td>07.92</td>
<td>81.16</td>
<td>58.33</td>
<td>58.33</td>
<td>11.25</td>
<td>40.84</td>
</tr>
<tr>
<td>$F$ vs. $T$</td>
<td>01.03</td>
<td>34.86</td>
<td>00.95</td>
<td>38.73</td>
<td>28.21</td>
<td>49.25</td>
<td>00.05</td>
<td>27.40</td>
</tr>
<tr>
<td>$S$ vs. $T$</td>
<td>06.55</td>
<td>55.02</td>
<td>05.81</td>
<td>58.25</td>
<td>51.19</td>
<td>60.04</td>
<td>05.28</td>
<td>35.07</td>
</tr>
<tr>
<td>$A$ vs. $T$</td>
<td>10.95</td>
<td>64.71</td>
<td>05.00</td>
<td>70.76</td>
<td>69.29</td>
<td>69.29</td>
<td>05.00</td>
<td>30.71</td>
</tr>
<tr>
<td>WDBC</td>
<td>55.67</td>
<td>37.26</td>
<td>×</td>
<td>×</td>
<td>37.26</td>
<td>45.82</td>
<td>47.04</td>
<td>45.84</td>
</tr>
</tbody>
</table>

3.4.3 Comparison of Generalization Error

This section presents results which compare the performance of MM-M, MM-R, MM-S and IC-SM when evaluated using the error measures $MajErr$ (3.19) and $OptErr$ (3.21).

Datasets: Experiments were done on the synthetic ($D_{10U}(10, 100, 100)$ and $D_{10B}(10, 100, 100)$) and real-world datasets described in Section 3.4.1.

Results: In all cases, the hyper-parameters ($C$ and/or $\epsilon$) for each classifier were tuned using a 3-fold cross-validation procedure. The results are summarized in Table 3.2. In case of synthetic datasets, the reported values represent the mean testset error achieved with the tuned hyper-parameters when trained with 10 different training sets each generated from the same synthetic data template. In case of the real-world datasets, the values represent cross-validation error with tuned hyper-parameters averaged over three cross-validation experiments. Hence the error values reported in the table represent a good estimate of the generalization error of the respective classifiers. Clearly, $MajErr$ and $OptErr$ are least for IC-SM; confirming that IC-SM achieves good generalization for interval data. Moreover, in case of many datasets, the proposed classifier outperforms the existing classifiers in terms of the $OptErr$ error measure.
Discussion of experimental results  The results on the synthetic experiments show that at all values of $\epsilon$, the classification margin with IC-SM is higher than that with MM-S. According to structural risk minimization principle of Vapnik [1], higher margin implies better generalization. The proposed classifier as expected achieves good generalization for interval data.

3.5 Summary

This chapter presents a novel maximum-margin linear classifier which achieves good generalization on data having interval-valued uncertainty. The key idea was to employ chance-constraints in order to handle uncertainty. The main contribution was to derive a tractable SOCP formulation, which is a safe approximation of the resulting CCP, using Bernstein schemes. The formulation ensures that, the probability of misclassification on interval data is less than a user specified upper-bound ($\epsilon$). Also, the geometric interpretation shows that the classifier views each training example as the region of intersection of its bounding hyper-rectangle and an ellipsoid centered at its mean.

The proposed linear classifier is robust to interval-valued uncertainty and is also not-overly-conservative. As shown in the chapter, this amounts to achieving higher classification margins and in turn better generalization than methods which employ the bounding hyper-rectangle information alone. As the results showed, the average error with the proposed classifier, in case of many synthetic and real-world datasets, is less than half of that with the existing methods.

The Bernstein relaxation schemes presented in this chapter not only aid in approximating the original CCP as a convex program, but also open avenues for efficient approximations of other CCP-based learning formulations (e.g. [8] and its variants). By employing rich partial information, the Bernstein schemes lead to less conservative relaxations. Hence exploitation of the Bernstein schemes in the context of learning is a good direction for research.
Chapter 4

Robustness to Bounded Moments

Abstract

This chapter studies the problem of constructing robust classifiers when the training set has interval-valued uncertainty, but here the uncertainty is modeled by employing partial information such as bounds of second order moments along with support information. This problem is also posed as a Chance-Constrained Program (CCP) and then relaxed by employing Bernstein bounding schemes. Prior to this work, only the Chebyshev based relaxations were exploited in learning theory using full covariance information. Bernstein bounds employ richer partial information and hence can be far less conservative than Chebyshev bounds. Due to this efficient modeling of uncertainty, the resulting classifiers are also robust to moment estimation errors. It achieves wider classification margins and hence better generalization than existing methods. Methodologies for classifying uncertain test datapoints and error measures for evaluating classifiers robust to uncertain data have been discussed thoroughly. Experimental results on synthetic and real-world datasets show that the proposed classifiers are better equipped to handle data uncertainty and outperform state-of-the-art in many cases.

4.1 Introduction

This chapter studies the problem of constructing robust classifiers when the training set has interval-valued uncertainty, but here the uncertainty is modeled by employing partial information such as bounds of second order moments (mean and variance) along with support information of the uncertain training datapoints. Recent studies have shown that classifiers which explicitly handle the uncertainty in training data, perform better than the classifiers which ignore such information [2, 3, 4]. In the past, robust classifiers which either employ support
information [5, 6] or second order moments of the noise distribution [7] were derived. Since these classifiers employ limited partial information i.e. either support or moment information alone, though they achieve robustness to uncertainty, they tend to be overly-conservative. However, as richer partial information is employed, uncertainty can be better modeled — leading to classifiers which are robust but not overly-conservative. As discussed at various stages of this chapter, a direct consequence of non-conservative modeling of the uncertainty is an increase in classification margin and hence an increase in generalization ability of the classifier.

Similar to Chapter 3, the idea here is also to derive a maximum-margin formulation which employs chance-constraints for the uncertain training datapoints. Each chance-constraint ensures that the corresponding uncertain training datapoint is classified correctly with high probability. Until now only the Chebyshev bounding schemes were employed to relax various CCP based learning formulations [8, 7, 9]. Here the Bernstein bounding schemes [10, 11] are employed for relaxing the resulting Chance-Constrained Program (CCP) as a Second Order Cone Program (SOCP). The Bernstein based schemes, by employing richer partial information (support and second order moment information), lead to less conservative modeling of the uncertainty than the Chebyshev based schemes, which employ moments information alone. Using SOCP relaxation, various maximum-margin formulations are derived which employ different levels of information about the uncertain datapoints. Table 4.1 summarizes the formulations\(^1\) derived in this chapter and the partial information employed by them. Please note that (MM-SM) use same information as used by (IC-SM) (3.16) in previous chapter. It is easy to prove that, they both are equivalent.

---

\(^1\)Similar to 3.1.1 nomenclature of formulations: prefix “MM” denotes Maximum Margin classifier. Partial information of Support, Mean, Variance employed by the classifier are denoted by ‘S’, ‘M’, ‘V’ respectively. The symbol ‘B’ denotes that the corresponding classifier employs bounds on moments rather than exact moments. The suffix ‘I’ indicates that the corresponding classifier is a variant, whose meaning will be clear later in the text. For example, the abbreviation “MM-SBMV” stands for a maximum-margin classifier which employs support, bounds on means and variances of uncertain datapoints.
Chapter 4. Robustness to Bounded Moments

Table 4.1: Table summarizing formulations presented in the chapter and the partial information employed by them.

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Support</th>
<th>$1^{st}$ Moment</th>
<th>$2^{nd}$ Moment</th>
<th>Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓</td>
<td>bounds</td>
<td>bounds</td>
<td>MM-SBMV (4.11), MM-SBMV-I (4.18)</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>exact</td>
<td>exact</td>
<td>MM-SMV (4.19), MM-SMV-I (4.21)</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>bounds</td>
<td>×</td>
<td>MM-SBM (4.22), MM-SBM-I (4.24)</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>exact</td>
<td>×</td>
<td>MM-SM (4.25), MM-SM-I (4.27)</td>
</tr>
</tbody>
</table>

4.1.1 Review of Past Work

The well known SVM (3.1) formulation [1], which assumes that the training datapoints are known exactly has already been discussed in Section 3.1 of previous chapter. The maximum-margin principle by employing the means (MM-M (3.2)), only support (MM-S (3.4)) have also been described in Section 3.1.1. In this section, past work done on maximum-margin classification with uncertain data using the means and covariances of the uncertain training datapoints are reviewed.

Chance constraints were previously used in handling uncertainty in the context of linear classifiers [5, 2, 7]. Using the means, $\mu_i$, and covariances, $\Sigma_i \equiv \text{cov}[x_i]$ of the uncertain training datapoints, and employing the Chebyshev’s inequality, classifiers which are robust to uncertainty have been derived [2, 7]. The formulation can be written as the following SOCP:

$$(\text{MM-MC}) \quad \min_{w,b,\xi_i} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \quad \text{s.t.} \quad y_i(w^T \mu_i + b) \geq 1 - \xi_i + \kappa_c \|\Sigma_i \frac{1}{2} w\|_2, \quad \xi_i \geq 0, \quad i = 1, \ldots, n \quad (4.1)$$

where $\kappa_c = \sqrt{\frac{1-\varepsilon}{\varepsilon}}$ and $\varepsilon \in [0, 1]$ is a user-given parameter (3.5). The robust formulations derived in [6] turn out to be special cases of the (MM-MC) formulation.

The formulation (MM-MC) is nearest in spirit to the present work. As shown in [7], (MM-MC) is the result of relaxing a CCP based formulation using the Chebyshev inequality. Relaxation schemes based on the Chebyshev’s inequality are known to be conservative as they employ second moment information alone. In this chapter, similar to previous chapter the
Bernstein bounding schemes are employed in order to relax the same CCP based maximum-margin formulation. The Bernstein based relaxation employs both the support and second order moment information and hence leads to less conservative modeling of the uncertainty, which as discussed above is the key in deriving classifiers with good generalization. Clearly, (MM-S) is the most robust method for handling uncertainty (provided the support of $x_i$ is correctly known). However it is very conservative, as it assumes the true example can lie anywhere in its bounding hyper-rectangle. On the other extreme, (MM-M) is least conservative, but is not robust to uncertainty as it assumes the true example to be the mean. The conservative nature of (MM-MC) is controlled by the parameter $\epsilon$. For example, when $\epsilon = 0$ (MM-MC) is the least conservative (in fact same as (MM-M)) and as $\epsilon \to 1$ the conservative nature of (MM-MC) increases. Note that, more conservative a method is, less is its margin, $\frac{2}{||w||^2}$. According to the structural risk minimization principle [1], more the margin is, better is its generalization ability. Hence it is important to design classifiers which are robust to uncertainty in training examples, as well as, not overly conservative. In this chapter, a maximum-margin formulation which employs both the support and bounds of second order moment information is derived. Since it uses richer partial information about the uncertain data, it is expected to be less conservative and hence achieve better generalization than the existing methods. As the proposed formulation requires the knowledge of bounds on second order moments rather than the exact moments, it is also robust to moment estimation errors.

4.1.2 Contribution

The key contribution of this chapter is to derive tractable maximum-margin formulations which employ both the support and the second order moment information of the uncertain datapoints in order to build the decision boundary. Since the proposed classifiers employ richer partial information and models the uncertainty better, they achieve better generalization than the existing methods. Also, the proposed classifiers require the knowledge of bounds on second order moments rather than the exact moments, which are often unknown. Thus, in addition of being robust to uncertainty and not being overly conservative, the proposed classifiers are also inherently robust to moment estimation errors.
Chapter 4. Robustness to Bounded Moments

4.1.3 Structure of the Chapter

The remainder of the chapter is organized as follows: Section 4.2 presents the main contribution of the chapter, a maximum-margin SOCP formulation which employs the support and bounds on the second order moments of the uncertain datapoints in order to achieve robustness. The section also presents various specializations of this formulation to the scenarios presented in Table 4.1. The subsequent section discusses the issue of classifying uncertain test datapoints and presents various error measures which evaluate the performance of classifiers which handle uncertain data. In Section 4.4, experimental results which compare the performance of the proposed methods and the existing methods are presented. The chapter is concluded in Section 4.5, by summarizing the work.

4.2 Maximum-margin Formulations for Uncertain Data

This section presents maximum-margin classification formulations which employ partial information about the uncertain training data in order to achieve robustness. Recall the SVM formulation (3.1), which is considered here as the baseline formulation. Now, since the datapoints \( x_i \) are uncertain or random, the constraints in (3.1) can no longer be satisfied deterministically. Hence, alternatively, the constraint can be satisfied with high probability. Therefore, it is required that the chance-constraints (3.5) are satisfied. The chance-constraints are restated as follows:

\[
\text{Prob} \left( y_i (w^T x_i + b) \leq 1 - \xi_i \right) \leq \epsilon \quad \forall i \in \{1, 2, \ldots, n\},
\]

(4.2)

where \( 0 \leq \epsilon \leq 1 \) is a user-given parameter close to 0, denoting an upper bound on the misclassification error made on \( x_i \). Thus, the chance-constraints (4.2) ensure that the uncertain datapoints are classified correctly with high probability \((1 - \epsilon)\). Using these chance-constraints, the following maximum-margin formulation, similar in spirit to SVMs, can be written:

\[
\min_{w, b, \xi} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} \quad \text{Prob} \left( y_i (w^T x_i + b) \leq 1 - \xi_i \right) \leq \epsilon, \quad \xi_i \geq 0, \quad i = 1, \ldots, n
\]

(4.3)
Chapter 4. Robustness to Bounded Moments

The above formulation is already defined in Chapter 3 at Section 3.6 and hence-forth will be denoted as (CCP) The (CCP) is hard to solve because the constraints are typically non-convex. In the remainder of the section several safe convex approximations of (CCP) are derived, depending on what information about the uncertain training datapoints, \( x_i \), is known. The notation used hence-forth is summarized below (\( \forall i = 1, \ldots, n \), \( j = 1, \ldots, d \)):

\[
\begin{align*}
a_{ij} \leq x_{ij} \leq b_{ij} & \quad a_i \equiv [a_{i1} \ldots a_{id}]^T \quad b_i \equiv [b_{i1} \ldots b_{id}]^T \\
n_{ij} = \frac{b_{ij} - a_{ij}}{2} & \quad \mu_i \equiv [\mu_{i1} \ldots \mu_{id}]^T \quad \mu_i^- \equiv [\mu_{i1}^- \ldots \mu_{id}^-]^T \quad \mu_i^+ \equiv [\mu_{i1}^+ \ldots \mu_{id}^+]^T \\
n_{ij} = \frac{b_{ij} + a_{ij}}{2} & \quad \mu_{ij} \equiv \mathbb{E}[x_{ij}] \in [\mu_{ij}^-, \mu_{ij}^+] \quad \text{var}[x_{ij}] \in [0, \sigma_{ij}^2] \\
L_i \equiv \text{diag}(\{l_{i1} \ldots l_{id}\}) & \quad m_i \equiv [m_{i1} \ldots m_{id}]^T \\
\hat{\mu}_{ij} \equiv \frac{\mu_{ij} - m_{ij}}{l_{ij}} & \quad \hat{\mu}_{ij}^- \equiv \frac{\mu_{ij}^- - m_{ij}}{l_{ij}} \quad \hat{\mu}_{ij}^+ \equiv \frac{\mu_{ij}^+ - m_{ij}}{l_{ij}} \quad (4.4)
\end{align*}
\]

4.2.1 Formulations using Support and Bounds on 2\textsuperscript{nd} Order Moments

This section presents a maximum-margin classification formulation which assumes bounds on means and variances, as well as support (bounding hyper-rectangles) of the uncertain training datapoints are known. It is also assumed that the features used to describe the data are independent — in other words, the random variables \( x_{ij} \), \( j = 1, \ldots, d \) are assumed to be independent.

The key idea is to derive convex constraints involving the above partial information, which when satisfied imply that the chance-constraints (4.2) are satisfied. To this end, the following theorem is presented, which specializes the Bernstein approximation schemes described in [10, 11]:

**Theorem 4.1.** Consider the notation in (4.4). Assuming partial information of support (\( a_{ij} \leq x_{ij} \leq b_{ij} \)), bounds on means (\( \mu_{ij}^- \leq \mu_{ij} \leq \mu_{ij}^+ \)) and variances (\( 0 \leq \text{var}[x_{ij}] \leq \sigma_{ij}^2 \)) of independent random variables \( x_{ij} \), \( j = 1, \ldots, d \) are known, the chance-constraint (4.2) is satisfied if the
following convex constraint in variables, \((w, b, \xi_i)\), holds:

\[
1 - \xi_i + y_i b + \sum_j \left( \max \left[ -y_i \mu_{ij}^- w_j, -y_i \mu_{ij}^+ w_j \right] \right) + \kappa \|L_i \hat{\Sigma}_i w\|_2 \leq 0 \tag{4.5}
\]

where

\[
\hat{\Sigma}_i = \text{diag} \left( [v \left( \hat{\mu}_{i1}^{-}, \hat{\mu}_{i1}^{+}, \hat{\sigma}_{i1} \right) \ldots v \left( \hat{\mu}_{in}^{-}, \hat{\mu}_{in}^{+}, \hat{\sigma}_{in} \right)] \right) \tag{4.6}
\]

and the function \(v(\hat{\mu}^{-}, \hat{\mu}^{+}, \hat{\sigma})\) is as defined in (C.4)

**Proof.** The chance-constraint (4.2) can be written as:

\[
\text{Prob} \left( u_i^T x_i + u_{i0} \geq 0 \right) \leq \varepsilon \tag{4.7}
\]

where \(u_{i0} = 1 - \xi_i + y_i b\) and \(u_i = -y_i w\).

Using Markov inequality and independence of random variables, \(x_{ij}, j = 1, \ldots, d\),

\[
\text{Prob} \left( u_i^T x_i + u_{i0} \geq 0 \right) \leq \exp \{ \zeta u_{i0} \} \prod_j \mathbb{E} \left[ \exp \{ \zeta u_{ij} x_{ij} \} \right], \forall \zeta \geq 0 \tag{4.8}
\]

The key of modeling chance constraint (4.7) now depends on how one upperbounds the moment generating functions, \(\mathbb{E} \left[ \exp \{ t x_{ij} \} \right], t \in \mathbb{R}\). To continue the proof, the following lemma\(^2\) will be used:

**Lemma 4.1.** Consider the notation in (4.4) and suppose the support and bounds on mean, variance for the random variable \(x_{ij}\) are known. Then,

\[
\mathbb{E} \left[ \exp \{ t x_{ij} \} \right] \leq \exp \left\{ v \left( \hat{\mu}_{ij}^{-}, \hat{\mu}_{ij}^{+}, \hat{\sigma}_{ij} \right)^2 \frac{l_{ij}^2}{2} + \max \left[ \mu_{ij}^{-} t, \mu_{ij}^{+} t \right] \right\} \forall t \in \mathbb{R} \tag{4.9}
\]

where \(v(\hat{\mu}^{-}, \hat{\mu}^{+}, \hat{\sigma})\) is as defined in (C.4).

\(^2\)Refer Appendix C.1 for proof of Lemma 4.1
Using Lemma 4.1 and (4.8) for \( \forall \zeta \geq 0 \):

\[
\log \left[ \text{Prob} \left( \mathbf{u}^\top \mathbf{x}_i + u_{i0} \geq 0 \right) \right] \leq \zeta \left( u_{i0} + \sum_j \left( \max \left[ -y_i \mu_{ij}^-, w_j, -y_i \mu_{ij}^+ w_j \right] \right) \right) + \frac{\zeta^2}{2} \| L_i \Sigma_i w \|_2^2
\]

Since this inequality holds for all values of \( \zeta \), if this can be ensured that for certain \( \zeta \) the right-hand side of the inequality is \( \leq \log(\varepsilon) \), then that would satisfy the chance-constraint (4.7) and have:

\[
\zeta \left( u_{i0} + \sum_j \left( \max \left[ -y_i \mu_{ij}^-, w_j, -y_i \mu_{ij}^+ w_j \right] \right) \right) + \frac{\zeta^2}{2} \| L_i \Sigma_i w \|_2^2 \leq \log \varepsilon \quad (4.10)
\]

In the case of \( q = 0 \), the above inequality is possible only if \( p < 0 \), \( \varepsilon \in [0, 1] \). Now suppose \( q > 0 \), then the value of \( \zeta \) needs to be found for which the LHS of (4.10) is minimized. This minimized value is 0 if \( p \geq 0 \) and \( \frac{-p^2}{2q^2} \) if \( p < 0 \). Again since \( \varepsilon \in [0, 1] \), \( p \geq 0 \) is not allowed. Substituting \( \frac{-p^2}{2q^2} \) in LHS of (4.10), one gets \( \frac{p^2}{4q} \leq \kappa^2 \leftrightarrow p + \kappa q \leq 0 \). Hence either in the case \( q = 0 \) or \( q > 0 \), \( p + \kappa q \leq 0 \) is the sufficient condition for satisfying the chance-constraint (4.2). Substituting the values of \( p, q, u_{i0} \) in this inequality, (4.5) is obtained. This completes the proof of Theorem 4.1.

Replacing the chance-constraints in (CCP) (4.3) with the deterministic (convex) constraints (4.5), a maximum-margin formulation, which ensures that the probability of misclassification when trained with uncertain data, \( \mathbf{x}_i \), is less than \( \varepsilon \), is obtained. This formulation can be written as the following SOCP:

\[
\text{(MM-SBMV)} \quad \min_{w, b, \xi_i, z_{ij}} \quad \frac{1}{2} \| w \|_2^2 + C \sum_{i=1}^n \xi_i \quad \text{s.t.} \quad 1 - \xi_i + y_i b + \sum_j z_{ij} + \kappa \| L_i \Sigma_i w \|_2 \leq 0, \quad z_{ij} \geq -y_i \mu_{ij}^- w_j, z_{ij} \geq -y_i \mu_{ij}^+ w_j, \quad \xi_i \geq 0 \quad (4.11)
\]

The values of \( \nu(\mu_{ij}^-, \mu_{ij}^+, \delta_{ij}) \) (C.4) can be calculated numerically (refer Appendix B.2). The
SOCP (MM-SBMV) can be efficiently solved using cone program solvers like SeDuMi\textsuperscript{3}, Mosek\textsuperscript{4} or CPLEX\textsuperscript{5}.

In the following, a geometrical interpretation of the formulation (MM-SBMV) is presented. To this end, consider the following lemma:

**Lemma 4.2.** Consider the notation in (4.4). Also, let the set

\[ E(\mu_i, \kappa L_i \tilde{\Sigma}_i) \equiv \{ x = \mu_i + \kappa L_i \tilde{\Sigma}_i u : \|u\|_2 \leq 1 \} \]

(4.12)

represents an ellipsoid centered at \( \mu_i \), whose shape and size are determined by \( \kappa L_i \tilde{\Sigma}_i \). Consider the problem of correctly classifying points belonging to the union of ellipsoids \( E(\mu_i, \kappa L_i \tilde{\Sigma}_i) \) over \( \mu_i \in [\mu_i^-, \mu_i^+] \):

\[ y_i (w^T x + b) \geq 1 - \xi_i, \ \forall \ x \in \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} E(\mu_i, \kappa L_i \tilde{\Sigma}_i) \]

(4.13)

The continuum of constraints (4.13) are satisfied if and only if (4.5) holds.

**Proof.** It appears to be:

\[ (4.13) \iff \max_{x \in \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]}} \left( -y_i w^T x \right) + 1 - \xi_i + y_i b \leq 0 \]

\[ \iff \max_{\mu_i \in [\mu_i^-, \mu_i^+], \|u\|_2 \leq 1} \left( -y_i w^T (\mu_i + \kappa L_i \tilde{\Sigma}_i u) \right) + 1 - \xi_i + y_i b \leq 0 \]

\[ \iff \max_{\mu_i \in [\mu_i^-, \mu_i^+]} \left( -y_i w^T \mu_i \right) + \max_{\|u\|_2 \leq 1} \left( -\kappa y_i w^T L_i \tilde{\Sigma}_i u \right) + 1 - \xi_i + y_i b \leq 0 \]

\[ \iff (4.5) \]

This completes the proof. \( \square \)

The above lemma shows that the formulation (MM-SBMV) views each uncertain training datapoint as the set \( \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+] \bigcup E(\mu_i, \kappa L_i \tilde{\Sigma}_i) } \) and does a maximum-margin classification using

---

3Available at http://sedumi.mcmaster.ca/
4Available at http://www.mosek.com/index.php?id=7
5Available at http://www.ilog.com/products/cplex/
these uncertainty sets.

Note that the size of the uncertainty set, and hence robustness and conservative nature of the classifier depend on $\kappa$ (and hence on $\varepsilon$). More specifically, as the upper bound on misclassification error, $\varepsilon$, decreases, size of the uncertainty set increases. However from the support information it is known that the true training datapoint can never lie outside its bounding hyper-rectangle denoted as:

$$\mathcal{R}(a_i, b_i) \equiv \{x : a_i \leq x \leq b_i\}$$ (4.14)

Thus it is possible to obtain less conservative classifiers by employing constraints using uncertainty sets as the intersection of $\bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{E}(\mu_i, \kappa L_i \tilde{\Sigma}_i)$ and the bounding hyper-rectangle (4.14). To this end the following lemma is presented:

**Lemma 4.3.** Consider the problem of correctly classifying points belonging to the set $\mathcal{R}(a_i, b_i) \cap \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{E}(\mu_i, \kappa L_i \tilde{\Sigma}_i)$:

$$y_i(w^T x + b) \geq 1 - \xi_i, \ \forall \ x \in \mathcal{R}(a_i, b_i) \cap \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{E}(\mu_i, \kappa L_i \tilde{\Sigma}_i)$$ (4.15)

The continuum of constraints (4.15) are satisfied if and only if the following convex constraint holds:

$$1 - \xi_i + y_i b + \sum_j \max \left[ -a_{ij} (y_i w_j + u_{ij}) + b_{ij} (y_i w_j + u_{ij}) \right] + \sum_j \left( \max \left[ \mu_{ij^-} u_{ij}, \mu_{ij^+} u_{ij} \right] \right) + \kappa \|L_i \tilde{\Sigma}_i u_i\|_2 \leq 0$$ (4.16)

**Proof.** The constraints (4.15) hold if and only if:

$$1 - \xi_i + y_i b + \max_{x \in \mathcal{R}(a_i, b_i) \cap \bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{E}(\mu_i, \kappa L_i \tilde{\Sigma}_i)} \left( -y_i w^T x \right) \leq 0$$

Note that, the term with max in the above inequality is nothing but the support function of the
Chapter 4. Robustness to Bounded Moments

set \( R(a_i, b_i) \cap \left( \bigcup_{\mu \in [\mu_i^-, \mu_i^+]} C(\mu, \kappa L_i \Sigma_i) \right) \) (denoted by \( I_{R(a_i, b_i)} \cap \left( \bigcup_{\mu \in [\mu_i^-, \mu_i^+]} C(\mu, \kappa L_i \Sigma_i) \right) \left( -y_i w \right) \)).

Since support function of intersection of two sets is the infimal convolution of support functions of the individual sets (see Section 16, [34]), and it turns out to be:

\[(4.15) \iff 1 - \xi_i + y_i b + \inf_{u_i + v_i = -y_i w} \left\{ I_{\bigcup_{\mu \in [\mu_i^-, \mu_i^+]} C(\mu, \kappa L_i \Sigma_i)} (u_i) + I_{R(a_i, b_i)} (v_i) \right\} \leq 0 \]

\[\iff \exists u_i, v_i \ni 1 - \xi_i + y_i b + \left\{ I_{\bigcup_{\mu \in [\mu_i^-, \mu_i^+]} C(\mu, \kappa L_i \Sigma_i)} (u_i) + I_{R(a_i, b_i)} (v_i) \right\} \leq 0, \]

\[u_i + v_i = -y_i w \quad (4.17)\]

Let the entries in vectors \( u_i, v_i \) be \( u_{ij}, v_{ij}, j = 1, \ldots, d \) respectively. Then by Lemma 4.2,

\[I_{\bigcup_{\mu \in [\mu_i^-, \mu_i^+]} C(\mu, \kappa L_i \Sigma_i)} (u_i) = \sum_j \left( \max \left[ \mu_j^- u_{ij}, \mu_j^+ u_{ij} \right] \right) + \kappa \| L_i \Sigma_i u_i \|_2 \]

. Also,

\[I_{R(a_i, b_i)} (v_i) = \sum_j \max \left[ a_{ij} v_{ij}, b_{ij} v_{ij} \right] \]

. Hence, (4.15) is satisfied if and only if:

\[1 - \xi_i + y_i b + \sum_j \max \left[ a_{ij} v_{ij}, b_{ij} v_{ij} \right] + \sum_j \left( \max \left[ \mu_j^- u_{ij}, \mu_j^+ u_{ij} \right] \right) + \kappa \| L_i \Sigma_i u_i \|_2 \leq 0 \]

and \( u_i + v_i = -y_i w \). Eliminating the variable \( v_i \) (4.16) is obtained. \( \square \)

Replacing the chance-constraints (4.3) in (CCP) with the convex constraint (4.16), a maximum-margin classification formulation is obtained. This formulation is robust to uncertain data as well as less conservative than the MM-SBMV formulation. This formulation can be written
as the following SOCP:

\[
\begin{align*}
\text{(MM-SBMV-I)} \quad \min_{w, b, \xi_i, z_{ij}, u_i} & \quad \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{subject to} & \quad 1 - \xi_i + y_i b + \sum_j z_{ij} + \sum_j z_{ij} + \kappa \| L_i \hat{S}_i u_i \|_2 \leq 0, \\
& \quad z_{ij} \geq \mu_{ij} u_{ij}, \quad z_{ij} \geq \mu_{ij} u_{ij}, \quad \xi_i \geq 0, \\
& \quad z_{ij} \geq -a_{ij}(y_i w + u_{ij}), \quad z_{ij} \geq +b_{ij}(y_i w + u_{ij})
\end{align*}
\] (4.18)

Note that, the proposed formulations (4.11) and (4.18) are not only robust to the uncertainties in data but are also robust towards moment estimation errors. This is because the formulations employ bounds on mean \((\mu_{ij}, \mu_{ij}^+)\) and variance \((\sigma_{ij}^2)\) rather than the true means and variances of the uncertain datapoints, which are often unknown.

In the special case where the moments of the training datapoints are known, one can use them as \(\mu_i = \mu_i^- = \mu_i^+\) and \(\text{var}[x_{ij}] = \sigma_{ij}^2\). And the formulation (4.11) can be written as:

\[
\begin{align*}
\text{(MM-SMV)} \quad \min_{w, b, \xi_i} & \quad \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{subject to} & \quad y_i (w^T \mu_i + b) \geq 1 - \xi_i + \kappa \| L_i \hat{S}_i w \|_2, \quad \xi_i \geq 0
\end{align*}
\] (4.19)

where
\[
\hat{S}_i = \text{diag}(\{\sqrt{\nu(\hat{\mu}_{i1}, \hat{\mu}_{i1}, \hat{\sigma}_{i1})} \ldots \sqrt{\nu(\hat{\mu}_{in}, \hat{\mu}_{in}, \hat{\sigma}_{in})}\})
\] (4.20)

Also, in this case, the formulation (4.18) can be written as:

\[
\begin{align*}
\text{(MM-SMV-I)} \quad \min_{w, b, \xi_i, u_i, z_{ij}} & \quad \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i \\
\text{subject to} & \quad 1 - \xi_i + y_i b + \sum_j z_{ij} + \mu_i^T u_i + \kappa \| L_i \hat{S}_i u_i \|_2 \leq 0, \\
& \quad z_{ij} \geq -a_{ij}(y_i w + u_{ij}), \quad z_{ij} \geq +b_{ij}(y_i w + u_{ij}), \quad \xi_i \geq 0
\end{align*}
\] (4.21)

Note that, the uncertainty sets associated with the formulations (4.19) and (4.21) are \(\varepsilon(\mu_i, \kappa L_i \hat{S}_i)\) and \(\mathcal{R}(a_i, b_i) \cap \varepsilon(\mu_i, \kappa L_i \hat{S}_i)\) respectively.
The formulations derived in this section employ partial information of both support and second order moments of uncertainty. These formulations can be specialized to cases where support and mean information alone are available. Though this increases the applicability of the formulations, the resulting classifiers are more conservative as they employ less information. The details are discussed in the subsequent section. Preliminary results (4.27) in this section have been presented earlier in the Chapter 3 as (3.16). The equivalence of both the formulation can be proved by replacing $z_{ij}$ of (3.16) by $v_{ij}l_{ij}$ and using (3.10).

4.2.2 Formulations using Support and Bounds on Means

This section presents a maximum-margin classification formulation which assumes that the bounds on means and the bounding hyper-rectangles (support) for the uncertain training datapoints are known. Note that, though no variance information is assumed in this case, the bounding hyper-rectangles imply a natural variance bound: $\text{var}[x_{ij}] = \text{var}[x_{ij} - m_{ij}] = E[x_{ij} - m_{ij}]^2 - (\mu_{ij} - m_{ij})^2 \leq l_{ij}^2 - (\mu_{ij} - m_{ij})^2 = \sigma^2_{ij}$. Now by (4.4), $\hat{\sigma}^2_{ij} = \frac{\sigma^2_{ij}}{\hat{\mu}^2_{ij}} + \hat{\mu}^2_{ij} = 1$. In other words, all the formulations presented in the previous section, where the variance information was assumed, can be specialized using $\hat{\sigma}_{ij} = 1$. Hence the formulation (4.11), in this case, reduces to the following SOCP:

$$\begin{align*}
\text{(MM-SBM)} & \min \quad \frac{1}{2} ||w||^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \quad 1 - \xi_i + y_i b + \sum_j z_{ij} + \kappa ||L_i \tilde{\Sigma}_i w||_2 \leq 0, \\
& \quad z_{ij} \geq -y_i \mu_{ij} w_j, \ z_{ij} \geq -y_i \mu_{ij}^+ w_j, \ \xi_i \geq 0
\end{align*}$$

(4.22)

where

$$\tilde{\Sigma}_i = \text{diag}([v(\hat{\mu}_{i1}^-), \hat{\mu}_{i1}^+, 1, \ldots, v(\hat{\mu}_{in}^-), \hat{\mu}_{in}^+, 1])$$

(4.23)
Chapter 4. Robustness to Bounded Moments

Also, formulation (4.18), which employs a less conservative uncertainty set than (4.11), reduces to:

\[
\text{(MM-SBM-I)} \quad \begin{array}{c}
\min_{w, b, \xi_i, z_{ij}, \tilde{z}_{ij}, i} \\
\frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n \xi_i \\
s.t. \\
1 - \xi_i + y_i b + \sum_j \tilde{z}_{ij} + \sum_j z_{ij} + \kappa \|L_i \tilde{\Sigma}_i u_i\|_2 \leq 0,
\end{array}
\]

\[ z_{ij} \geq \mu_{ij}^- u_{ij}, \quad z_{ij} \geq \mu_{ij}^+ u_{ij}, \quad \xi_i \geq 0,
\]

\[ \tilde{z}_{ij} \geq -a_{ij} (y_i w_j + u_{ij}), \quad \tilde{z}_{ij} \geq +b_{ij} (y_i w_j + u_{ij}) \]

\[
(4.24)
\]

The uncertainty sets associated with the formulations (4.22) and (4.24) are \([\hat{m}_i^- \hat{m}_i^+] \in \mathcal{R}(\mu_i, \kappa L_i \tilde{\Sigma}_i)\) and \(\mathcal{R}(\mathbf{a}_i, b_i) \cap \left( \bigcup_{\mu_i \in [\mu_i^- \mu_i^+]} \mathcal{R}(\mu_i, \kappa L_i \tilde{\Sigma}_i) \right)\) respectively.

It is interesting to note that, the value of \(v(\hat{\mu}^-, \hat{\mu}^+, 1)\) can be computed analytically in the case where \(\hat{\mu}^- \leq 0, \hat{\mu}^+ \geq 0\). This case occurs for e.g. when the mean, \(\mu_i\), lies somewhere around the mid-point of its bounding hyper-rectangle. In particular, if the noise distribution of the \(i^{th}\) uncertain datapoint is symmetric, this assumption is trivially true. To this end, consider the following lemma:

**Lemma 4.4.** For any \(\hat{\mu}^- \in [-1, 0], \hat{\mu}^+ \in [0, 1]\), the following is true:

\[
v(\hat{\mu}^-, \hat{\mu}^+, 1) = \sqrt{1 - (\hat{\mu}_{\text{min}})^2}
\]

where \(\hat{\mu}_{\text{min}} = \min(|\hat{\mu}^-|, |\hat{\mu}^+|)\).

Again, in the special case where the means \(\mu_i\) are known, one has \(\mu_i = \mu_i^- = \mu_i^+\). Using this, the formulation (4.22) reduces to the following SOCP:

\[
\text{(MM-SM)} \quad \begin{array}{c}
\min_{w, b, \xi_i} \\
\frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n \xi_i \\
s.t. \\
y_i (\mathbf{w}^\top \mu_i + b) \geq 1 - \xi_i + \kappa \|L_i \tilde{\Sigma}_i w\|_2, \quad \xi_i \geq 0
\end{array}
\]

\[
(4.25)
\]

\[\text{Refer Appendix C.2 for a proof of Lemma 4.4.}\]
where

$$\Sigma_i = \text{diag}([\mu_i; \mu_{i1}, 1] \ldots [\mu_{in}; \mu_{in}, 1])$$ (4.26)

Also, in this case, formulation (4.24) reduces to:

$$\begin{align*}
\text{(MM-SM-I)} & \min_{w, b, \xi, \bar{u}_i, \bar{z}_{ij}} \frac{1}{2} ||w||^2 + C \sum_{i=1}^{n} \xi_i \\
\text{s.t.} & \\
& 1 - \xi_i + y_i b + \sum_{j} \bar{z}_{ij} + \mu_i \bar{u}_i + \xi ||L_i \Sigma_i u_i||_2 \leq 0, \\
& \bar{z}_{ij} \geq -a_{ij}(y_i w_j + u_{ij}), \bar{z}_{ij} \geq +b_{ij}(y_i w_j + u_{ij}), \xi_i \geq 0
\end{align*}$$ (4.27)

Note that, the uncertainty sets with the formulations (4.25) and (4.27) are $\mathcal{E}(\mu_i, \kappa L_i \Sigma_i)$ and $\mathcal{B}(a_i, b_i) \cap \mathcal{E}(\mu_i, \kappa L_i \Sigma_i)$ respectively. It is also interesting to note that in the special cases $\epsilon = 1$ and $\epsilon = 0$, the formulation (MM-SM-I) degenerates to (MM-M) and (MM-S) formulations respectively. A comparison of the proposed formulations based on their conservative nature is presented in the subsequent section.

### 4.2.3 Note on the Conservative Nature of the Various Formulations

This section summarizes the formulations presented in the chapter and provides a comparison based on their conservative nature. The formulations presented in the chapter (refer Table 4.1 also) can be categorized based on whether they employ:

- **First order or second order moment information:** Formulations named using the symbol ‘V’ employ variance (second order moment) information. In general, formulations which employ second order moments are less conservative than those which employ first order moments.

- **Bounds or exact moment information:** Formulations named using the symbol ‘B’ employ bounds on moments rather than the exact moments. In general, formulations which employ moment bounds are more conservative than the ones which use exact moments. This is because, they guard themselves against moment estimation errors also. However, they are more pragmatic, as in case of real-world data the exact moments
are never known.

- **Ellipsoidal or intersection of ellipsoidal and hyper-rectangular uncertainty sets:**
  Formulations suffixed with the symbol ‘I’ employ uncertainty sets which are intersections of ellipsoidal and hyper-rectangular sets. By construction, these formulations are less conservative than their counterparts using ellipsoidal uncertainty sets.

The empirical results in Section 4.4.2 support the comparisons presented here. Hence the proposed formulations and in particular MM-SMV-I, lead to robust but not overly-conservative classifiers. Formulations like MM-SBMV-I which employ bounds on moments increase the practical applicability of the proposed methodology as the true moments are never known exactly.

### 4.3 Prediction rules and Error metrics

This section discusses the issue of classifying uncertain test datapoints and presents various error measures for evaluating the performance of classifiers which are robust to uncertain data. Different label prediction strategies can be employed based on the level of information available regarding the uncertainty in test datapoints. Table 4.2 summarizes the various forms in which the uncertainty in datapoints can be represented (here, $y_{pr}$ denotes the predicted label; refer (4.4) for further notation). For each of them, the partial information available and the corresponding label prediction methodologies are also indicated. The applicability of a particular classification formulation presented in the chapter can be decided based on the partial information available (refer Table 4.1 also). As noted earlier, even in the cases where either support or moment information is not available at all (e.g. Form 1, 2 in Table 4.2), formulation (MM-SM-I) can be applied with $\varepsilon = 1$ and $\varepsilon = 0$ respectively. Please note that, all the methodologies summarized in Table 4.2 actually try to consider the predicted label of the majority portion of the uncertain region as predicted label for test example. Hence, once a suitable labeling strategy is chosen, the Majority error, $\text{MajErr}$, can be calculated as percentage of wrongly
Table 4.2: Table summarizing the possible ways of representing uncertainty in data. Corresponding classification formulations and label prediction strategies are indicated.

<table>
<thead>
<tr>
<th>Form</th>
<th>Representation</th>
<th>Partial Information</th>
<th>Labeling Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Single datapoint d</td>
<td>Mean</td>
<td>$y^{pr} = \text{sign}(w^T d + b)$</td>
</tr>
<tr>
<td>2</td>
<td>Intervals</td>
<td>Support</td>
<td>$y^{pr} = \text{sign}(w^T \mu_i + b)$</td>
</tr>
<tr>
<td>3</td>
<td>Intervals and moments</td>
<td>Support, Moments</td>
<td>$y^{pr} = \text{sign}(w^T \mu_i + b)$</td>
</tr>
<tr>
<td>4</td>
<td>Replicates $d_1, d_2, \ldots$</td>
<td>Support, Moments</td>
<td>$y^{pr}$ is majority label of replicates Label of replicate: $\text{sign}(w^T d_i + b)$</td>
</tr>
</tbody>
</table>

Classified test datapoints:

$$\text{MajErr} = \frac{\sum_{i=1}^{n} I[y_i^{pr} \neq y_i]}{\# \text{test datapoints}} \times 100$$  \hspace{1cm} (4.28)

In cases where the information of both support and moments are available, interesting error measures can be derived based on the discussion presented in Section 4.2.1. Suppose an uncertain test datapoint $x_i$ with label $y_i$ is given and let $\mu_{ij}^{opt} = \arg \max_{\mu \in [\mu_i^-; \mu_i^+]} (-y_i \mu w_j)$. Then by Theorem 4.1, the true probability of misclassification of the test datapoint $x_i$ will be less than or equal to $\varepsilon$ (i.e. $\text{Prob} [y_i(w^T x_i + b) \leq 0] \leq \varepsilon$) if:

$$y_i(w^T \mu_i^{opt} + b) \geq \sqrt{2 \log(1/\varepsilon) w^T L_i^2 \Sigma_i w}$$

The above inequality is obtained at by setting $1 - \xi_i$ term to zero in (4.5) and then re-arranging the terms. Using this, one can calculate the least value of $\varepsilon = \varepsilon_{opt}$ for which the above inequality is satisfied:

$$\varepsilon_{opt} = \exp \left\{ \frac{-(w^T \mu_i^{opt} + b)^2}{2(w^T L_i^2 \Sigma_i w)} \right\}$$

Please note that, by Lemma 4.2, $\varepsilon_{opt}$ is the value of $\varepsilon$ for which the uncertainty region

$$\bigcup_{\mu_i \in [\mu_i^-; \mu_i^+]} C(\mu_i, \kappa L_i \Sigma_i)$$

touches the discriminating hyperplane, $w^T x + b = 0$. By the very definition of $\varepsilon_{opt}$, the true probability of misclassification of the test datapoint $x_i$ will be less than or equal to it. This
Table 4.3: Table showing $\epsilon_{opt}$ for different error measures $\text{OptErr}^{1-4}$

<table>
<thead>
<tr>
<th>Error Measure</th>
<th>Formulation</th>
<th>Uncertainty Region</th>
<th>$\epsilon_{opt}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr$^1_i$</td>
<td>MM-SBMV</td>
<td>$\bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{G}(\mu_i, \kappa L_i \Sigma_i)$</td>
<td>$\exp\left{ \frac{(w^T \mu_i^p + b)^2}{2(w^T L_i^2 \Sigma_i^w)} \right}$</td>
</tr>
<tr>
<td>OptErr$^2_i$</td>
<td>MM-SMV</td>
<td>$\mathcal{G}(\mu_i, \kappa L_i \Sigma_i)$</td>
<td>$\exp\left{ \frac{-(w^T \mu_i + b)^2}{2(w^T L_i^2 \Sigma_i^w)} \right}$</td>
</tr>
<tr>
<td>OptErr$^3_i$</td>
<td>MM-SBM</td>
<td>$\bigcup_{\mu_i \in [\mu_i^-, \mu_i^+]} \mathcal{G}(\mu_i, \kappa L_i \Sigma_i)$</td>
<td>$\exp\left{ \frac{(w^T \mu_i^p + b)^2}{2(w^T L_i^2 \Sigma_i^w)} \right}$</td>
</tr>
<tr>
<td>OptErr$^4_i$</td>
<td>MM-SM</td>
<td>$\mathcal{G}(\mu_i, \kappa L_i \Sigma_i)$</td>
<td>$\exp\left{ \frac{-(w^T \mu_i + b)^2}{2(w^T L_i^2 \Sigma_i^w)} \right}$</td>
</tr>
<tr>
<td>OptErr$^c_i$</td>
<td>MM-MC</td>
<td>$\mathcal{G}(\mu_i, \kappa \Sigma_i^2)$</td>
<td>$\frac{w^T \Sigma_i w}{(w^T \mu_i + b)^2 + w^T \Sigma_i w}$</td>
</tr>
</tbody>
</table>

leads to the following error definition on each test datapoint:

$$
\text{OptErr}_i = \begin{cases} 
1 & \text{if } y_i \neq y_i^p \\
\epsilon_{opt} & \text{if } y_i = y_i^p \text{ and } \exists x \in \mathcal{R}(a_i, b_i) \ni y_i(w^T x + b) < 0 \\
0 & \text{if } y_i(w^T x + b) \geq 0 \forall x \in \mathcal{R}(a_i, b_i)
\end{cases} \quad (4.29)
$$

Now the analysis in the previous paragraph can be repeated with different uncertainty regions derived in this chapter. The error measures with different uncertainty regions differ only in computing the value of $\epsilon_{opt}$. The uncertainty regions, the corresponding error measures and $\epsilon_{opt}$ are summarized in Table 4.3. An error measure derived using the Chebyshev based relaxation is also presented in the table (denoted by OptErr$^c_i$). The overall error, OptErr$^{1-4,c}$, can be calculated as percentage of OptErr$^{1-4,c}_i$ over all test datapoints:

$$
\text{OptErr}^{1-4,c} = \frac{\sum_i \text{OptErr}^{1-4,c}_i}{\# \text{test datapoints}} \times 100 \quad (4.30)
$$

Note that, both MajErr and OptErr$^{1-4,c}$ can be estimated for any hyperplane classifier, provided the partial information employed is available. Also, since the OptErr$^{1-4,c}$ error measures are derived using the Bernstein/Chebyshev probability bounds, they can be thought of as the worst-case error estimates or as the upper bounds on the “true” MajErr (which is the MajErr estimated on a very large set of test datapoints). Experimental results reported in the
subsequent section show that the proposed classifiers achieve lower \( \text{MajErr} \) and \( \text{OptErr}^{1-4,c} \) when compared to existing uncertain data classifiers.

### 4.4 Experimental Evaluation

This section presents experimental results which extensively compare the proposed (refer Table 4.1) and existing methodologies (refer Section 4.1.1) for classifying uncertain data.

In particular it would be interesting to explore the following questions.

1. Comparison of the conservative nature of the various robust classification constraints presented in Table 4.4

2. Comparative study of the classification margin achieved by the various robust classification presented in this chapter (see Table 4.4)

3. Generalization performance of the various robust classification presented in Table 4.4 along with \( \text{MM-R} \)

4. Robustness to moment estimation errors of the \( \text{MM-SM-I}, \text{MM-SMV-I}, \text{MM-SBM-I}, \text{MM-SBMV-I} \), against \( \text{MM-M}, \text{MM-R}, \text{MM-S}, \text{MM-MC} \)

As the key motivation is to develop robust as well as non-overly-conservative classifiers, the first set of experiments, presented in Section 4.4.2, compare the conservative nature of various robust classification constraints derived in the chapter and existing in the literature. In particular, these experiments compare the conservative nature of Chebyshev and Bernstein bounding schemes for relaxing CCP based learning formulations. As experimental results show, Bernstein schemes lead to far less conservative relaxations than Chebyshev schemes and hence have potential to be exploited in building robust classifiers for uncertain data. Section 4.4.3 presents experiments which compare the margin, \( 2/\|w\|_2 \), achieved by the proposed and existing robust classifiers on synthetic datasets. The results show that the proposed classifiers achieve higher margin and hence have potential to generalize better.

Section 4.4.4 presents the key empirical results of the chapter — comparison of various robust classifiers discussed in this chapter using the error measures \( \text{MajErr} \) (4.28) and
OptErr\(^{1-4,c}\) (4.30). Results show that in case of all datasets, the proposed classifiers achieve better generalization than state-of-the-art.

As mentioned earlier, classifiers derived using Bernstein relaxation schemes are also inherently robust to moment estimation errors. This is because the proposed classifiers require knowledge of moment bounds rather than the exact moments themselves. Section 4.4.5 presents experiments comparing the robustness of various uncertain data classifiers to moment estimation errors. The results show that the proposed classifiers are less susceptible to moment estimation errors than existing classifiers.

### 4.4.1 Datasets

Experiments were performed on synthetic datasets and two real-world datasets

**WBCD** The data set has been described in Section 3.4.1. For each training data the mean, standard-error and maximum values of the 10 features are provided and from this information the support and moments for all features of a datapoint are estimated. Bounds on the means, \([\mu^-, \mu^+]\), are estimated using the standard-error information. This dataset is an example of Form-3 data (refer Table 4.2).

**Micro-array** This Dataset also has been used and described in the previous chapter (see Section 3.4.1).

**Synthetic** Synthetic datasets were generated by following the methodology described in Section 3.4.1. For this chapter \(Dd_U(S, N, L)\), \(Dd_B(S, N, L)\), \(Dd_{SN}(S, N, L)\) and \(Dd_{ST}(S, N, L)\) were generated and the distribution used for generating replicates were Uniform, truncated Beta, skew-Normal and skew-t noise distributions respectively. In each case, synthetic test data was also generated independently.

Both the **Micro-array** and **Synthetic** datasets stand as examples of Form-4 data (refer Table 4.2). Also, in these cases, the support and moments for each datapoint were estimated from the corresponding replicates. The bounds on mean \((\mu^-, \mu^+)\) and bounds on variance \((\sigma^2)\) were estimated using the Hotelling’s \(T^2\)-statistic (see e.g. page 227, [35]) and Cochran’s theorem (see e.g. page 419, [36]) respectively.
Chapter 4. Robustness to Bounded Moments

Platform and package: All the proposed formulation listed in the Table 4.1 and MM-MC have been implemented in Matlab with the help of Sedumi\textsuperscript{7} [33]. MM-S has also been implemented in Matlab with help of standard QP solver quadprog\textsuperscript{8}. libSVM\textsuperscript{9} has been used as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.00 GHz processors and 16GB of RAM.

4.4.2 Comparison of the Conservative Nature of the Various Robust Constraints

This section compares the conservative nature of the various robust classification constraints presented in this chapter. In particular, conservativeness of the various convex relaxations of the chance-constraint \( \text{Prob}(w^T x_i + b \leq 0) \leq \varepsilon \) have been compared here. Note that, this constraint is a variant of the original chance-constraint (4.2) with \( y_i = 1 \) and the \( 1 - \xi_i \) term neglected. Table 4.4 summarizes various relaxations of this chance-constraint derived using the Bernstein and Chebyshev bounding schemes. Also, the constraints (3.2) in (MM-M) and (3.4) in (MM-S) are accordingly modified and are shown in the Table 4.4. These represent the two extreme relaxations — most lenient and most conservative. The relaxations which use bounds on moments are not compared here in order to have a fair comparison with Chebyshev schemes — which can only be employed if exact moments are known. Constraints shown in the table for MM-SMV-I and MM-SM-I can be derived easily from Lemma 4.3, rather than from constraints in the corresponding formulations (4.21) and (4.27).

Now, suppose the value of \( w \) is fixed. Then, the conservative nature of the various relaxations can be compared by looking at the least upper bound on \( b \). Greater the value of the least upper bound on \( b \) is, lesser is the conservativeness of the corresponding relaxation. Noting this observation, the following experiment was done: in each run of the experiment a random vector \( w \) was chosen and datapoints were sampled from a random distribution. Random distributions were simulated by employing random parameters for the truncated skewed

\textsuperscript{7}http://sedumi.ie.lehigh.edu/
\textsuperscript{8}http://www.mathworks.com/access/helpdesk/help/toolbox/optim/ug/quadprog.html
\textsuperscript{9}http://www.csie.ntu.edu.tw/ cjlin/libsvm/
t-distribution. Using the sampled datapoints, support and second order moments were estimated. Employing this partial information and the chosen value of \( w \), the least upper bound on \( b \) with various constraints can be calculated using Table 4.4.

**Datasets:** The synthetic data \( \mathbf{D}^{10_U(50, 100, 100)} \) were used.

**Results:** Now let the value of \( b \) thus obtained with MM-M and MM-S be \( b_m \) and \( b_s \) respectively. Figure 4.1 shows the plot of \( b_{\text{norm}} = \frac{b - b_s}{b_m - b_s} \) averaged over 50 such experimental runs at different values of \( \varepsilon \). Since \( \varepsilon \) is a small number denoting upper bound on misclassification probability, only values of \( \varepsilon \in [0, 0.2] \) are interesting and hence are shown in the Figure 4.1. Note that the value of \( b_{\text{norm}} \) with MM-SMV-I and MM-SMV is the highest — proving that the Bernstein relaxation schemes are less conservative than the Chebyshev based schemes. It is interesting to note that for \( \varepsilon \leq 0.04 \), even the Bernstein relaxations using first order moments ((4.25) in MM-SM and (4.27) in MM-SM-I) are less conservative than the second order moment based Chebyshev relaxations ((4.1) in MM-MC). Hence, formulations derived using the proposed methodology model uncertainty in a less conservative fashion and are expected to achieve better generalization while being robust to uncertainties in data.

### 4.4.3 Comparison of Classification Margin

This section presents experimental results which compare various robust formulations based on the classification margin \( (2/\|w\|_2) \) achieved by them.

<table>
<thead>
<tr>
<th>Relaxation Scheme</th>
<th>Formulation</th>
<th>Relaxed Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bernstein bounding</td>
<td>MM-SMV (4.19)</td>
<td>( b \leq w^\top \mu_i - \kappa |L_i \Sigma_i w|_2 )</td>
</tr>
<tr>
<td>Bernstein bounding</td>
<td>MM-SMV-I (4.21)</td>
<td>( b \leq \max(\langle w^\top \mu_i - \kappa |L_i \Sigma_i w|_2, w^\top m_i - |L_i w|_1) )</td>
</tr>
<tr>
<td>Bernstein bounding</td>
<td>MM-SM (4.25)</td>
<td>( b \leq w^\top \mu_i - \kappa |L_i \Sigma_i w|_2 )</td>
</tr>
<tr>
<td>Bernstein bounding</td>
<td>MM-SM-I (4.27)</td>
<td>( b \leq \max(\langle w^\top \mu_i - \kappa |L_i \Sigma_i w|_2, w^\top m_i - |L_i w|_1) )</td>
</tr>
<tr>
<td>Chebyshev bounding</td>
<td>MM-MC (4.1)</td>
<td>( b \leq w^\top \mu_i - \kappa |\Sigma_j^2 w|_2 )</td>
</tr>
<tr>
<td>Mean based</td>
<td>MM-M (3.2)</td>
<td>( b \leq w^\top \mu_i )</td>
</tr>
<tr>
<td>Support based</td>
<td>MM-S (3.4)</td>
<td>( b \leq w^\top m_i - |L_i w|_1 )</td>
</tr>
</tbody>
</table>
Datasets: The synthetic datasets $D_{2U}(50, 100, 100)$, $D_{10U}(50, 100, 100)$, $D_{2B}(50, 100, 100)$, $D_{10B}(50, 100, 100)$, $D_{2SN}(50, 100, 100)$ and $D_{10SN}(50, 100, 100)$ were created to do experiments reported in this section and a fixed value of $C = 10$ is used on each dataset.

Results: Figure 4.4.3 shows plots of margin vs. $\varepsilon$ with various formulations at a fixed value of $C = 10$ on different synthetic datasets. Clearly, the margins achieved by the proposed formulations at all $\varepsilon$ values are higher than those achieved with MM-MC and MM-S. Since higher margins imply better generalization, the proposed classifiers are expected to generalize better. Also note that for higher dimensional datasets the gain in margin with the proposed classifiers is higher. The plots also show that the proposed formulations model uncertainty in a less conservative fashion, regardless of the underlying noise distribution. This is expected as the proposed methodology does not make any distributional assumptions. The margin achieved by MM-M is highest as it neglects the uncertainty of the datapoints and assumes mean as the only possible position for the datapoint. Also, the trends shown in Figure 4.4.3 remained the
same at different values of the $C$ parameter.

### 4.4.4 Comparison of Generalization Error

This section presents the experimental results comparing OptErr$^{1-4,\epsilon}$ (4.30) and MajErr (4.28) incurred by the various robust classifiers presented in this chapter. In cases where the uncertainty in datasets is represented using replicates (e.g. Micro-array data), the comparison is also done with an SVM constructed assuming each replicate as a training datapoint. Henceforth this classifier is denoted as MM-R.

**Datasets:** Only real world dataset WBCD and Micro-array datasets were used here.

**Results:** The results are summarized in Table 4.5. In each case, the hyper-parameters ($C$ and/or $\epsilon$) were tuned using a 5-fold cross-validation procedure. The reported error values represent the cross-validation error obtained using the corresponding tuned set of hyper-parameters averaged over three 5-fold cross-validation experiments. Hence lower the values of OptErr$^{1-4,\epsilon}$ and MajErr is, better is the generalization ability of the corresponding robust classifier. Clearly, the proposed classifiers incurred the least error on all the datasets. Moreover in terms of the OptErr$^{1-4,\epsilon}$ measures, the proposed classifiers outperformed state-of-the-art in case of most of the datasets.

The results also show that the nearest competitor to the proposed classifiers is the MM-MC classifier — which is also based on chance-constrained techniques. The results hence show that, in general, formulations based on chance-constrained techniques and moreover the formulations relaxed efficiently using the Bernstein bounds are best suited for handling uncertainties in data. As mentioned earlier, the classifiers derived using the Bernstein relaxations require knowledge of bounds on moments performs better rather than the exact moments.

Clearly, the results in Table 4.5 show that the variants which employ bounds on moments achieve better generalization than the ones which employ exact moments. This inherent advantage of the proposed methodology is again illustrated in the subsequent section.
Figure 4.2: Figure comparing margins achieved by 8 formulation at various $\epsilon$ values ($D_{2B}(50,100,100)$ on top-left, $D_{10B}(50,100,100)$ on top-right, $D_{2U}(50,100,100)$ on middle-left, $D_{10U}(50,100,100)$ on middle-right, $D_{2SN}(50,100,100)$ on bottom-left, $D_{10SN}(50,100,100)$ on bottom-right)
Table 4.5: Table comparing classification error ($100 \times \text{ErrorMeasure}\%$) in terms of $\text{OptErr}^{1-4,c}$ and $\text{MajErr}$ incurred with various robust classifiers using real world dataset – WBCD and Micro-array.

<table>
<thead>
<tr>
<th>Dataset:</th>
<th>WBCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr$^1$</td>
<td>37.26</td>
</tr>
<tr>
<td>OptErr$^2$</td>
<td>37.26</td>
</tr>
<tr>
<td>OptErr$^3$</td>
<td>37.26</td>
</tr>
<tr>
<td>OptErr$^4$</td>
<td>37.26</td>
</tr>
<tr>
<td>OptErr$^c$</td>
<td>37.26</td>
</tr>
<tr>
<td>MajErr</td>
<td>55.67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset:</th>
<th>A vs. F</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr$^1$</td>
<td>41.57</td>
</tr>
<tr>
<td>OptErr$^2$</td>
<td>39.21</td>
</tr>
<tr>
<td>OptErr$^3$</td>
<td>46.13</td>
</tr>
<tr>
<td>OptErr$^4$</td>
<td>46.00</td>
</tr>
<tr>
<td>OptErr$^c$</td>
<td>77.18</td>
</tr>
<tr>
<td>MajErr</td>
<td><strong>00.00</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset:</th>
<th>A vs. S</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr$^1$</td>
<td>61.10</td>
</tr>
<tr>
<td>OptErr$^2$</td>
<td>59.25</td>
</tr>
<tr>
<td>OptErr$^3$</td>
<td>64.74</td>
</tr>
<tr>
<td>OptErr$^4$</td>
<td>64.64</td>
</tr>
<tr>
<td>OptErr$^c$</td>
<td>86.24</td>
</tr>
</tbody>
</table>

*continued on next page*
Table 4.5 — continued from previous page

<table>
<thead>
<tr>
<th>Dataset: ( \mathcal{A} ) vs. ( \mathcal{I} )</th>
<th>MM-M</th>
<th>MM-R</th>
<th>MM-S</th>
<th>MM-MC</th>
<th>MM-SM-I</th>
<th>MM-SMV-I</th>
<th>MM-SBM-I</th>
<th>MM-SBMV-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>MajErr</td>
<td>09.02</td>
<td>08.65</td>
<td>61.69</td>
<td>06.10</td>
<td>06.10</td>
<td>5.71</td>
<td>06.10</td>
<td>06.52</td>
</tr>
<tr>
<td>OptErr1</td>
<td>67.33</td>
<td>75.00</td>
<td>58.33</td>
<td>54.37</td>
<td>53.95</td>
<td>53.95</td>
<td>47.22</td>
<td>50.74</td>
</tr>
<tr>
<td>OptErr2</td>
<td>65.54</td>
<td>73.32</td>
<td>58.33</td>
<td>52.97</td>
<td>52.68</td>
<td>52.68</td>
<td>46.77</td>
<td>50.08</td>
</tr>
<tr>
<td>OptErr3</td>
<td>70.61</td>
<td>78.03</td>
<td>58.33</td>
<td>56.89</td>
<td>56.37</td>
<td>56.37</td>
<td>47.22</td>
<td>51.67</td>
</tr>
<tr>
<td>OptErr4</td>
<td>70.51</td>
<td>77.94</td>
<td>58.33</td>
<td>56.81</td>
<td>56.28</td>
<td>56.28</td>
<td>47.22</td>
<td>51.67</td>
</tr>
<tr>
<td>OptErrC</td>
<td>88.94</td>
<td>93.23</td>
<td>58.33</td>
<td>79.33</td>
<td>58.33</td>
<td>58.33</td>
<td>47.22</td>
<td>51.67</td>
</tr>
<tr>
<td>MajErr</td>
<td>09.72</td>
<td><strong>06.67</strong></td>
<td>58.33</td>
<td>08.33</td>
<td>08.33</td>
<td>09.72</td>
<td>08.89</td>
<td>11.11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset: ( \mathcal{F} ) vs. ( \mathcal{I} )</th>
<th>MM-M</th>
<th>MM-R</th>
<th>MM-S</th>
<th>MM-MC</th>
<th>MM-SM-I</th>
<th>MM-SMV-I</th>
<th>MM-SBM-I</th>
<th>MM-SBMV-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr1</td>
<td>31.23</td>
<td>35.21</td>
<td>47.68</td>
<td>24.52</td>
<td>24.76</td>
<td>24.76</td>
<td><strong>22.03</strong></td>
<td>22.15</td>
</tr>
<tr>
<td>OptErr2</td>
<td>29.27</td>
<td>33.32</td>
<td>46.89</td>
<td>22.71</td>
<td>22.96</td>
<td>22.96</td>
<td><strong>20.24</strong></td>
<td>20.38</td>
</tr>
<tr>
<td>OptErr3</td>
<td>34.99</td>
<td>38.86</td>
<td>49.31</td>
<td>28.17</td>
<td>28.39</td>
<td>28.39</td>
<td><strong>25.70</strong></td>
<td>25.79</td>
</tr>
<tr>
<td>OptErr4</td>
<td>34.86</td>
<td>38.73</td>
<td>49.25</td>
<td>28.05</td>
<td>28.27</td>
<td>28.27</td>
<td><strong>25.57</strong></td>
<td>25.66</td>
</tr>
<tr>
<td>OptErrC</td>
<td>69.91</td>
<td>73.70</td>
<td><strong>62.95</strong></td>
<td>66.51</td>
<td>62.95</td>
<td>62.95</td>
<td><strong>62.95</strong></td>
<td>62.95</td>
</tr>
<tr>
<td>MajErr</td>
<td>01.03</td>
<td>00.95</td>
<td>28.21</td>
<td><strong>00.00</strong></td>
<td><strong>00.00</strong></td>
<td><strong>00.00</strong></td>
<td>00.99</td>
<td>00.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset: ( \mathcal{F} ) vs. ( \mathcal{I} )</th>
<th>MM-M</th>
<th>MM-R</th>
<th>MM-S</th>
<th>MM-MC</th>
<th>MM-SM-I</th>
<th>MM-SMV-I</th>
<th>MM-SBM-I</th>
<th>MM-SBMV-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>OptErr1</td>
<td>51.69</td>
<td>54.70</td>
<td>63.89</td>
<td>41.08</td>
<td>41.45</td>
<td>41.45</td>
<td><strong>36.54</strong></td>
<td>36.88</td>
</tr>
<tr>
<td>OptErr2</td>
<td>49.84</td>
<td>52.81</td>
<td>63.87</td>
<td>39.63</td>
<td>40.07</td>
<td>40.07</td>
<td><strong>35.48</strong></td>
<td>35.79</td>
</tr>
<tr>
<td>OptErr3</td>
<td>55.14</td>
<td>58.37</td>
<td>63.89</td>
<td>43.76</td>
<td>44.03</td>
<td>44.03</td>
<td><strong>38.51</strong></td>
<td>38.88</td>
</tr>
<tr>
<td>OptErr4</td>
<td>55.02</td>
<td>58.25</td>
<td>63.89</td>
<td>43.66</td>
<td>43.94</td>
<td>43.94</td>
<td><strong>38.43</strong></td>
<td>38.80</td>
</tr>
<tr>
<td>OptErrC</td>
<td>79.84</td>
<td>83.52</td>
<td>63.89</td>
<td>71.16</td>
<td>63.89</td>
<td>63.89</td>
<td><strong>60.34</strong></td>
<td>60.78</td>
</tr>
<tr>
<td>MajErr</td>
<td>06.55</td>
<td><strong>03.81</strong></td>
<td>58.17</td>
<td>05.63</td>
<td>05.28</td>
<td>05.28</td>
<td>06.75</td>
<td>06.11</td>
</tr>
</tbody>
</table>

continued on next page
### 4.4.5 Robustness to Moment Estimation Errors

In this section, experiments comparing various formulations for robustness to moment estimation errors were presented.

**Datasets:** For training $D_{10ST}(10, 100, 100)$, $D_{15ST}(10, 100, 100)$ and $D_{20ST}(10, 100, 100)$ are generated. Please note that, 10 datasets for each template (e.g. $D_{10ST}$) differ only in terms of replicates; nominal datapoints are the same. Independently a test set ($D_{10ST}(1, 100, 1)$, $D_{15ST}(1, 100, 1)$ and $D_{20ST}(1, 100, 1)$) consisting of nominal datapoints alone is also generated. Hence the test set is not noisy and represents the true data. Various classifiers are trained (with fixed values of hyper-parameters $C = 10$ and $\epsilon = 0.4$) using each of the 10 training sets, and **MajErr** on the testsets was noted.

**Results:** The standard deviation in the test set error incurred on few synthetic data templates is summarized in Table 4.6. Ideally, since each of the 10 training sets represent the same “true” (nominal) set of datapoints, the variation in the test set accuracy must be zero. The results show the variation in test set error is the least for **MM-SBMV-I** and **MM-SBM-I** — which are the variants employing bounds on moments. This proves that the proposed classifiers are robust to moment estimation errors and illustrates the benefit of the proposed methodology. It is also
Chapter 4. Robustness to Bounded Moments

interesting to note that, the variation in test set accuracies with MM-SMV-I and MM-SMI is less than that with MM-R and MM-S. As mentioned earlier, the inherent advantage of the Bernstein relaxation is that it requires knowledge of bounds on moments than the exact moments. A consequence of this was better generalization, as illustrated in the results of the previous section.

Table 4.6: Table reporting standard deviation in testset error incurred by various robust classifiers.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D10ST</td>
<td>0.7160</td>
<td>0.8551</td>
<td>0.6779</td>
<td>0.7738</td>
<td>0.5996</td>
<td><strong>0.5389</strong></td>
<td>0.5446</td>
<td></td>
</tr>
<tr>
<td>D15ST</td>
<td>0.4908</td>
<td>0.4698</td>
<td>0.1829</td>
<td>0.4118</td>
<td>0.2222</td>
<td>0.2833</td>
<td>0.2042</td>
<td><strong>0.1504</strong></td>
</tr>
<tr>
<td>D20ST</td>
<td>0.8553</td>
<td>0.4517</td>
<td>0.2396</td>
<td>0.5864</td>
<td>0.2081</td>
<td>0.4286</td>
<td><strong>0.1853</strong></td>
<td>0.2086</td>
</tr>
</tbody>
</table>

Discussion of experimental results  In general, the Bernstein relaxation schemes are less conservative than the Chebyshev based schemes. This key feature was exploited in the proposed methodology for developing classifiers that model data uncertainty very efficiently. In future, it would be interesting to explore the applicability of Bernstein schemes for relaxing various other CCP-based learning formulations. Classifiers developed using the proposed methodology achieve higher margins and hence better generalization than state-of-the-art. Formulation using bounds on moments (MM-SBM-I, MM-SBMV-I) not only achieve good generalization but are less susceptible to moment estimation errors.

4.5 Summary

A novel methodology for constructing robust classifiers by employing partial information of support and moments of the uncertain training datapoints have been presented. The idea was to pose the uncertain data classification problem as a CCP and relax it as a convex SOCP formulation using Bernstein bounding schemes. The key advantage of the Bernstein relaxation scheme is to model uncertainty in a less conservative manner. Moreover, since the relaxation requires the knowledge of bounds on moments rather than the exact moments themselves, the resulting classifiers are also inherently robust to moment estimation errors. Using the
proposed methodology, various robust formulations employing different levels of partial information were derived. Interesting error measures for evaluating performance of classifiers robust to uncertain data were also presented. The performance of the proposed classifiers was empirically evaluated on various synthetic and real-world datasets.
Chapter 5

Robust Formulations for Handling Uncertain Kernel Matrices:
A Chance Constraint Approach

Abstract

In this Chapter, the problem of uncertainty in a Kernel matrix in the SVM framework is studied. It discusses a chance-constraint based setup when the uncertainty in the kernel matrix is modeled as independent noise over the kernel entries. A novel distribution free large deviation inequality is derived and when is applied to the problem at hand, leads to a co-positive program. Although such formulations are NP hard, under several cases of interest the problem simplifies to a convex program. The problem of uncertainty in a kernel matrix is motivated from the real world problem of classifying proteins when the structures are provided with some uncertainty. The formulation here naturally incorporates such uncertainties in structures in a principled manner leading to significant improvements over the state of the art.

5.1 Introduction

Given a dataset \( \mathcal{D} = \{(x_i, y_i) | x_i \in \mathcal{X}, y_i \in \{1, -1\}, i = 1, \ldots, n\} \) the Support Vector Machine(SVM) formulation \([1]\) defines a classifier of the form

\[
f(x) = \text{sign} \left( \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) \right),
\]

(5.1)
Chapter 5. Handling Uncertainty in Kernel Matrices

The kernel function, \( K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} \), plays the role of a dot product in an associated reproducing kernel Hilbert space [26]. Depending on the choice of this function, one can construct extremely powerful classifiers with non-linear decision boundaries which have proven to be extremely useful in practice. When one uses the kernel function, \( K(x_i, x) = x_i^T x \), one recovers the linear classifier. The coefficients, \( \alpha \), are determined by the following optimization problem,

\[
\max_{\alpha \in S_n, t_1} \alpha^T e - \frac{1}{2} t_1 \quad \text{s.t.} \quad \alpha^T Y K Y \alpha \leq t_1 \tag{5.2}
\]

where \( S_n = \{ \alpha | 0 \leq \alpha_i \leq C, \sum_{i=1}^{n} \alpha_i y_i = 0 \} \) and \( Y = \text{diag}(y_i) \). Each entry of the matrix \( K \), is defined by \( K_{ij} = K(x_i, x_j) \). Kernel based classifiers are extremely powerful as they can define complicated decision boundaries via the kernel functions and are becoming increasingly popular.

As discussed earlier, in many real-world problems there is inherent uncertainty in observations \( x \) including missing value problem. This chapter explores Chance-Constrained program for designing kernel classifiers (5.1) when observations are uncertain. More precisely, it is about the problem of designing robust classifiers when there is uncertainty in kernel matrix \( K \). To the best of knowledge acquired, there is no such study of this important problem in the existing literature.

The impact of uncertainty in individual example is treated as an additive uncertainty \( Z \). Consider the following chance constraint setting:

\[
\max_{t_1, \alpha \in S_n} \alpha^T e - \frac{1}{2} t_1 \quad \text{s.t.} \quad \alpha^T Y (K + Z) Y \alpha \leq t_1 \tag{5.3}
\]

\[
\text{Prob} \left( \alpha^T Y (K + Z) Y \alpha \leq t_1 \right) \geq 1 - \varepsilon \tag{5.4}
\]

where \( \varepsilon < 0.5 \). In this setting the inequality (5.4) ensures that the event \( \alpha^T Y (K + Z) Y \alpha \leq t_1 \) holds with high probability \((1 - \varepsilon)\) for any instantiation of the random variate \( Z \). It is assumed that \( K \) is a specified kernel matrix. Optimization problems involving chance constraints often turn out to be NP-hard and are actively studied by robust optimization community [10, 11].
An immediate motivation for studying this problem is the task of automated protein structure classification (see (1.2)). [15, 16] designed kernels based on similarity scores, like Root-Mean-Square-Deviation (RMSD) obtained from structural alignment algorithms e.g. DALI[17]. Existing methods assume that protein structures are determined exactly, without any uncertainty. However in reality, coordinates of atoms of protein structures are determined with uncertainty, governed by the resolution of X-ray diffraction experiment \(^1\). When the uncertainty becomes comparable to RMSD then the similarity scores become unreliable.

### 5.1.1 Review of Past Work

Chance constraints were previously used in handling uncertainty in the context of linear classifiers [5, 2, 7]. Assuming a full knowledge of covariance structure of the data uncertainty and using Chebychev inequality they [2, 7] formulated the problem as a Second Order Cone Program(SOCP). Instead of using a full covariance matrix, which is difficult to estimate, an alternative based only on the support information was proposed in [5]. The methods discussed in the literature do not readily apply to the case of arbitrary kernel functions and merits further study. In addition, studies presented in the previous chapters (please see Chapter 3 and 4) have been limited to linear classifiers. However, the application of these methods to (5.4) is not straightforward and requires further investigation.

### 5.1.2 Contribution

The idea is to solve (5.3) assuming that \(Z_{ij}\) are centered and independent of each other. Following two cases are studied here, a.) \(Z_{ij}\) is Gaussian, and b.) \(Z_{ij}\) has finite support. In case of Gaussian distribution, a novel formulation is derived which can be interpreted as a robust version of SVM. A major contribution of this chapter is a novel large deviation inequality which is applied to the finite support case. Using this inequality, a formulation similar to the Gaussian case for solving (5.3) is derived. The optimization problem in a general setting turns out to be

\(^1\)http://www.rcsb.org/pdb/
an instance of co-positive program which is NP hard in general. However, under certain assumptions the problem can be solved as a convex conic quadratic. It is interesting to note that under the i.i.d assumption the formulation reduces to SVM with a modified kernel function. Extensive experiments on synthetic datasets show that the current formulation is more robust than the standard SVM. The formulation developed here incorporates resolution information available in protein structures in a principled way yielding substantially better and experimental results show significant improvements in classification accuracy over existing state of the art methods and their obvious extensions. In addition, as expected, the new classifiers are more robust than the existing ones.

5.1.3 Organization

The chapter is organized as follows: the main contributions are described in Section 5.2. Section 5.3 presents algorithms to solve such problems, and Section 5.4.2 discusses metrics for measuring the performance of resultant classifiers. Section 5.5 reports experimental results. Section 5.6 summarizes the work and concludes the chapter.

5.2 Robust Formulations for Handling Uncertainty in Kernel Matrices

This section studies (5.4) when $Z_{ij}$ is independent across all $i, j$ with zero mean. It begins the study by assuming that the entries are Gaussian distributed and the Section 5.2.2 considers the more general case when the entries have finite support. To this end, a large deviation inequality on the inner product of a matrix with $Z$ is derived and that is later used in (5.4).

5.2.1 Uncertainty in Kernel matrix is Gaussian distributed

This section begins with the following Lemma.
Chapter 5. Handling Uncertainty in Kernel Matrices

**Lemma 5.1.** Let $Z$ be an $n \times n$ random matrix with entries $Z_{ij}$ independently distributed as $Z_{ij} \sim N(0, \sigma_{ij}^2)$. For every $W, A \in \mathbb{R}^{n \times n}$ the constraint

$$\text{Prob}(\text{Tr}\{(Z + W)A\} \geq t_1) \leq \varepsilon$$

is satisfied if the following holds.

$$\text{Tr}(WA) \leq t_1 + \Phi^{-1}(\varepsilon)\|\hat{\Sigma} * A\|_F$$

where $\hat{\Sigma}_{ij} = \sigma_{ij}$.

**Proof.** Observe that $\text{Tr}(ZA) \sim N(0, \|\hat{\Sigma} * A\|_F^2)$. This is true because $\text{Tr}(ZA)$ can be written as a weighted sum of independently distributed normal random variables. Using a standardized normal random variate, $u \sim N(0, 1)$, one can write $\text{Tr}(ZA) = u\|\hat{\Sigma} * A\|_F$. Using the CDF of $u$, defined as $\text{Prob}(u \leq u) = \Phi(u) = \int_{-\infty}^{u} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$ we get,

$$\text{Prob}(\text{Tr}\{(Z + W)A\} \geq t_1)$$

$$= \text{Prob}\left(u\|\hat{\Sigma} * A\|_F \geq t_1 - \text{Tr}(WA)\right)$$

$$= \Phi(-u), \text{ where } u = \frac{t_1 - \text{Tr}(WA)}{\|\hat{\Sigma} * A\|_F}$$

The second equality follows because $u$ is standard normal. This derivation shows that equation (5.5) can be restated as $\Phi(-u) \leq \varepsilon$ Noting that $\Phi$ is an increasing function of its argument one obtains $u \geq -\Phi^{-1}(\varepsilon)$ where $\Phi^{-1}$ is the inverse function of $\Phi$. Substituting the value of $u$ completes the proof of the theorem.

A direct application of the above lemma leads to the following theorem, which is the first result of the chapter.

**Theorem 5.1.** Let $Z$ be an $n \times n$ matrix whose entries are independently distributed with entries $Z_{ij} \sim N(0, \sigma_{ij}^2)$. Let $K = \overline{K} + Z$ be a noise corrupted matrix where $\overline{K}$ is an $n \times n$ known kernel matrix. For such a $K$ the constraint (5.4) in the formulation (5.3) is satisfied if the following

$^2K$ may not be PSD
Chapter 5. Handling Uncertainty in Kernel Matrices

holds.

\[ \sum_{i,j} y_i y_j \alpha_i \alpha_j K_{ij} - \Phi^{-1}(\varepsilon) \| \Sigma^{1/2} (\alpha \alpha^T) \|_F \leq t_1 \]  

(5.8)

Proof. Substituting \( W = \bar{K} \) and \( A_{ij} = \alpha_i y_i \alpha_j y_j \) in Lemma 5.1 proves the theorem.

This theorem points to a deterministic equivalent of the problem stated in (5.3). Note that for cases of interest \( \varepsilon < 0.5 \), which implies that \( \Phi^{-1}(\varepsilon) < 0 \). In a later section algorithmic approaches for solving such programs are discussed.

5.2.2 Uncertainty with Finite Support

In this section, the case of finite supported uncertainty in the kernel entries is discussed. A novel large deviation inequality is stated, proved and also exploited to obtain a deterministic constraint similar to the one obtained in the Gaussian case (see Theorem 5.1).

It begins by proving a novel large deviation inequality related to Bernstein scheme,

**Theorem 5.2.** Let \( Z \) be an \( n \times n \) random matrix with entries \( Z_{ij} \) independently distributed along with finite support, \( \text{Prob}(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1 \), and \( E(Z_{ij}) = 0 \). For every \( V \in \mathbb{R}^{n \times n} \), and \( u > 0 \)

\[ \text{Prob}(Tr(ZV) \geq u) \leq \exp \left\{ -\frac{1}{2} \frac{u^2}{\|V\|_F^2} \right\} \]  

(5.9)

\( l_{ij} = \frac{b_{ij} - a_{ij}}{2}, \ m_{ij} = \frac{b_{ij} + a_{ij}}{2}, \ \hat{\mu}_{ij} = \frac{m_{ij}}{l_{ij}}, \ \hat{\beta}_{ij} = l_{ij} \nu(\hat{\mu}_{ij}) \) where \( \nu(\hat{\mu}_{ij}) \) was already defined in (3.8) and restated as:

\[ \nu(\hat{\mu}_{ij}) = \min \left\{ \nu \geq 0 \left| \frac{\nu^2}{2} t^2 + \hat{\mu}_{ij} t - \log \left( \cosh(t) + \hat{\mu}_{ij} \sinh(t) \right) \geq 0, \forall t \in \mathbb{R} \right. \right\} \]  

(5.10)

Proof. As a consequence of Markov inequality and independence of entries of \( Z \), the following holds \( \forall \zeta \geq 0 \).

\[ \text{Prob}(Tr(ZV) \geq u) \leq e^{(-\zeta u)} \prod_{i,j} E \left( e^{(\zeta V_{ij} Z_{ij})} \right) \]  

(5.11)

Exploiting the convexity of the function, \( e^x \), one can upperbound the moment generating function of \( Z_{ij} \). More importantly for any \( t \in \mathbb{R} \) the following is true.
where, \( f_{ij}(\tilde{t}) = \log (\cosh(\tilde{t}) + \mu_{ij} \sinh(\tilde{t})) \). By using Taylor expansion around 0 the bound can be obtained as \( f_{ij}(\tilde{t}) \leq \frac{1}{2} t^2 + \mu_{ij} \tilde{t} \), which could be further tightened by considering \( f_{ij}(\tilde{t}) \leq \frac{1}{2} v(\mu_{ij})^2 \tilde{t}^2 + \mu_{ij} \tilde{t} \) where \( v(\mu_{ij}) \) is given in (5.10).

Substituting \( \tilde{t} = tl_{ij} \) the bound (5.12) can be written as

\[
\log \left( \frac{E(e^{Z_{ij}})}{e^{m_{ij} + f_{ij}(tl_{ij})}} \right) = \frac{1}{2} u^2 \left\| \beta' \ast V \right\|_F^2 - \frac{1}{2} \frac{u^2}{\left\| \beta' \ast V \right\|_F}
\]

The proof is completed by noting that minimization is attained at \( \tilde{\zeta} = \frac{u}{\left\| \beta' \ast V \right\|_F} \), obtained by solving an univariate quadratic optimization problem.

The values \( v(\mu_{ij}) \) (5.10) can be calculated numerically and is presented in Appendix B.1.

The inequality is of independent interest and a further study in this direction can be done in future. Instead, in this Thesis the inequality is applied to the problem at hand. Consider the following theorem

**Theorem 5.3.** Let \( Z \) be an \( n \times n \) matrix whose entries are independently distributed, given that \( P(a_{ij} \leq Z_{ij} \leq b_{ij}) = 1 \) and \( E(Z_{ij}) = 0 \). Let \( K = K + Z \) be a noise corrupted matrix where \( K \) is \( n \times n \) specified kernel matrix. For such a \( K \) the constraint (5.4) in formulation (5.3) is satisfied if the following holds.

\[
\sum_{ij} y_i y_j \alpha_i \alpha_j K_{ij} + \sqrt{2 \log(1/\epsilon)} \sqrt{\sum_{ij} \beta_{ij}^2} \leq t_1
\]

where \( \beta_{ij} = \beta_{ij}^2 \) and \( \beta_{ij}^2 \) is defined as in Theorem 5.2.

**Proof.** the proof begins by noting that the constraint (5.4) can be stated as

\[
Prob(Tr\{(Z + K)V\} \geq t_1) \leq \epsilon
\]
where $V_{ij} = \alpha_i y_i \alpha_j y_j$. A necessary condition for satisfying the above inequality can be obtained by exploiting the large deviation inequality stated in Theorem 5.2. In particular a direct application of the bound yields the following constraint

$$\sqrt{2\log(1/\epsilon)}\|\beta^* V \|_F \leq t_1 - Tr(KV)$$

(5.15)

Substituting $V$ in the above equation proves the theorem.

\[ \square \]

### 5.2.3 A Deterministic Optimization Problem

In light of the Theorem 5.1 and Theorem 5.3 one can motivate the following deterministic counterpart of (5.3).

\[
\begin{align*}
\text{(RSVM)} & \quad \min_{t_1, \alpha \in \mathbb{R}^n} \frac{1}{2} t_1 - \sum_i \alpha_i \\
& \quad \text{s.t. } \sum_{ij} y_i y_j \alpha_i \alpha_j K_{ij} + \kappa \sqrt{\sum_{ij} \beta_{ij} \alpha_i^2 \alpha_j^2} \leq t_1
\end{align*}
\]

(5.16)

where, $\kappa = \sqrt{2\log(1/\epsilon)}$ when uncertainty has finite support. This formulation is robust to uncertainty in the kernel entries and will be referred as Robust SVM (RSVM). In case of Gaussian uncertainty, $\kappa = -\Phi^{-1}(\epsilon)$ and $\beta_{ij} = \sigma^2_{ij}$ and this formulation will be referred as Robust SVM for Gaussian distribution (RSVM$^{(g)}$).

### 5.3 Algorithms for Solving the Robust Formulation

Few algorithms for solving (5.16) are discussed in this section. In general these problems (5.16) are instances of co-positive programs, which are extremely difficult (NP-hard) to solve [37]. While in several cases of interest the problems can be reduced to convex conic quadratic programs.

**Case 1 - $\beta$ is rank one:** For this case the formulation is equivalent of solving SVM. It is interesting to note that this case arises when the uncertainties in all entries of a kernel matrix
are i.i.d.

**Theorem 5.4.** Let $\beta$ be a rank one matrix, i.e. $\beta_{ij} = \rho_i \rho_j$ where $\rho$ is a vector with non-negative components. The formulation (5.16) is equivalent to a SVM with kernel $\overline{K} + \kappa \text{diag}(\rho)$.

**Proof.** Noting that, $\sqrt{\sum_{i,j} \beta_{ij} \alpha_i \alpha_j} = \sum_i \rho_i \alpha_i^2$ and eliminating $t_1$ the result follows. By design $\beta_{ij} \geq 0$ which implies that $\rho_i > 0$ [38].

The corresponding formulation is denoted by $\text{RSVM}_{\text{QP}}$ ($\text{RSVM}_{\text{QP}(g)}$ for Gaussian uncertainty).

As a corollary to the previous theorem one can prove following

**Theorem 5.5.** Let entries of the random matrix $Z$ be i.i.d. and $K = \overline{K} + Z$. In such a case formulation (5.16) is equivalent of solving a SVM with Kernel matrix $K = \overline{K} + \rho' \kappa I$ where $\rho' = \sqrt{\beta_{ij}} \; \forall \; i \; \text{and} \; j$.

**Proof.** Note that, for i.i.d. case $\rho_i = \rho'$ in the previous theorem and the proof follows.

**Case 2 - $\beta$ is psd:** If the matrix $\beta$ is positive semi-definite then the formulation can be posed as SOCP. To this end consider the following theorem.

**Theorem 5.6.** If both $K, \beta$ are symmetric positive-semi-definite matrices then the following formulation is equivalent to formulation (5.16).

\[
\begin{align*}
\min_{t_1, \alpha, t_2, \alpha} & \quad \frac{1}{2} t_1 - \sum_i \alpha_i \\
\text{s.t.} & \quad \kappa \| \beta^{1/2} \mathbf{v} \| \leq t_1 - t_2 \\
& \quad \|Y(\overline{K})^{1/2} \alpha\|_2^2 \leq t_2 \\
& \quad \alpha_i^2 \leq \vartheta_i
\end{align*}
\]

(5.17)

**Proof.** As $\beta, \overline{K}$ are positive-semi-definite matrices their matrix square roots, $\beta^{1/2}, \overline{K}^{1/2}$, exist. Considering the fact that, at optimality $\alpha_i^2 = \vartheta_i$ the theorem follows.

Note that, this formulation is second order cone representable and hence can be solved as an Second Order Cone Program (SOCP). This will be denoted by $\text{RSVM}_{\text{SOCP}}$ ($\text{RSVM}_{\text{SOCP}(g)}$ for Gaussian uncertainty).
Case 3- The case of general $\beta$: The case of general $\beta$ ($\beta_{ij} \geq 0$), is an instance of co-positive program. This Thesis has not studied this setting in detail here, but has proposed a general descent algorithm to solve this. In particular, it has used a modified Newton Method \[39\] with square penalty function leading to the following unconstrained approximation of (5.16).

$$\min_{\alpha} \mathcal{L}(\alpha) \left( = f(\alpha) + \frac{P}{2} \left( \sum_i \alpha_i y_i \right)^2 + \sum_{i: \alpha_i > C} (\alpha_i - C)^2 + \sum_{i: \alpha_i < 0} \alpha_i^2 \right) \tag{5.18}$$

where, 

$$f(\alpha) = \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j K_{ij} + \frac{1}{2} \kappa \sqrt{\sum_{i,j} \beta_{ij} \alpha_i^2 \alpha_j^2 - \sum_i \alpha_i} \text{ and } P \text{ is a user defined cost for penalty function.}$$

$\mathcal{L}$ was minimized with the Quasi Newton type method with DFP updation as follows

$$\alpha^{t+1} = \alpha^t - \gamma^t H^t \nabla_\alpha \mathcal{L} \text{ where, } \eta^t \text{ is step size, } H^t \text{ is approximate inverse of Hessian of } \mathcal{L} \text{ obtained by DFP procedure, and } \nabla_\alpha \mathcal{L} \text{ is the gradient of } \mathcal{L} \text{ w.r.t } \alpha.$$

This will be referred as RSVM\_QN (RSVM\_QN\_r for for Gaussian uncertainty). The algorithm suffers from the problem of local minimum. To alleviate that problem multiple starting points were used.

### 5.4 Prediction rules and Error metrics

The problem of attaching a label on an uncertain kernel is not yet well understood. Measures for evaluating the prediction scheme is also not clear. This section discusses the prediction rule and propose several performance measures.

**Choice of bias** The decision function for the classifier can be expressed as

$$f(\mathbf{x}) = \text{sign} \left( \sum_{i \in SV} y_i \alpha_i \mathbf{K}_i + \text{bias} \right)$$

where $SV$ is an index set of support vectors and $\mathbf{K}_i = K(\mathbf{x}_i, \mathbf{x})$ where $\mathbf{x}_i$ is the $i^{th}$ support vector.

In order to get robust performance and to reduce the effect of uncertainty in kernel, the bias
can be computed as
\[
bias = \frac{1}{\# SV} \left[ \sum_{j \in SV} y_j - \sum_{i,j \in SV} y_i \alpha_i \tilde{K}_{ij} \right]
\].

5.4.1 Prediction Rules

For each test data, \( x_t \) and all the training data points \( x_i \), the mean \( \tilde{K}_{ti} \) of the kernel entry \( K_{ti} \), and either variance \( \sigma^2_{ti} \) (in Gaussian case) or range \([a_{ti}, b_{ti}]\) (finite support case) for the uncertainty are available. In both cases, testing is done by generating multiple samples \( K'_{ti}, r = 1, \ldots, R \), for each kernel entry \( K_{ti} \), where \( R \) is the number of samples.

For each of these test values, \( \kappa = 0 \) (5.16) and \( \tilde{K}_{ti} = K'_{ti} \) were used. Hence, the decision function obtained from equation (5.16) can be written as:
\[
y'_{t} = \text{sign}\left( \sum_{j \in SV} \alpha_j y_j K'_{tj} + bias \right)
\]
One way to assign a label \( y_{t}^{pr} \) to \( x_t \) is by majority votes from \( y'_{t}; r = 1, \ldots, R \) So,
\[
y_{t}^{pr} = \text{sign}\left( \sum_{r=1}^{R} y'_{t} \right). \tag{5.19}
\]

5.4.2 Error Metrics

Let \( \mathcal{D}_{test} = \{(x_t, y_t) | t = 1, \ldots, n_{tst}\} \) be a test dataset on which the performance of RSVM is likely to be measured when (5.19) is used for prediction. For each test-point multiple instances of the kernel values are available. In such a case the choice of performance measure is not clear. To this end the following error measures have been defined.

The overall classification error can be calculated as:
\[
\text{MajorityErr} (ME) : \frac{\sum_{t=1}^{n_{tst}} 1(y_{t}^{pr} \neq y_t)}{n_{tst}} \tag{5.20}
\]
where, \( y_t \) is the true label for \( x_t \). The above measure is intuitive for measuring the goodness
of the classifiers to label a data point with uncertain kernel. However a robust classifier is expected to classify all the \( R \) instances of each data point \( \mathbf{x} \), correctly. To capture this notion of robustness, we propose another error measure (\textbf{RobustErr}) which counts the fraction of data points in \( \mathcal{D}_{\text{tst}} \) for which all the samples are classified correctly.

\[
\text{RobustErr}(RE) : \frac{\sum_{t=1}^{n_{\text{tst}}} 1(\exists r | y_r^t \neq y_t)}{n_{\text{tst}}}
\]

Apart from measuring the robustness, the performance of various models without considering any uncertainty in the test data, i.e. performance of various models only considering \( \mathbf{K} \) have been measured using the following prediction rule:

\[
\tilde{y}_t^{pr} = \text{sign} \left( \sum_i y_i \alpha_i K(\mathbf{x}_t, \mathbf{x}_i) + \text{bias} \right)
\]

A related performance measure, \textbf{NominalErr}, is defined as follows:

\[
\text{NominalErr}(NE) : \frac{\sum_{t=1}^{n_{\text{tst}}} 1(\tilde{y}_t^{pr} \neq y_t)}{n_{\text{tst}}}
\]

One can also treat each of the samples generated from uncertain data points as individual data points, and define a standard classification error (\textbf{SampleErr}), as:

\[
\text{SampleErr}(SE) : \frac{\sum_{t=1}^{n_{\text{tst}}} \sum_{r=1}^{R} 1(y_r^t \neq y_t)}{n_{\text{tst}}R}
\]

The following section reports experimental results for the techniques developed here and state-of-the-art methods with respect to the above mentioned metrics.

5.5 Experimental Evaluation

This section presents experimental results which compare the proposed RSVM, \( \text{RSVM}^{(g)} \), (5.16) and nominal-SVM (SVM with specified kernel) in terms of accuracy and robustness on the task of binary classification.
In particular it would be interesting to explore the following questions.

1. Comparison of RSVM and $RSVM(g)$ against the non-robust Nominal SVM

2. Comparison of robustness of the RSVM, $RSVM(g)$ and Nominal SVM formulation when faced with an uncertainty of type described in $\epsilon$

3. Verification of bounds stated in Lemma 5.1 and Theorem 5.2

The results for both synthetic data and resolution aware protein structure classification problem are reported here. The results demonstrate that the proposed formulations outperform state of the art techniques with respect to both traditional error measures and new metrics defined in section 5.4.2.

5.5.1 Datasets

To perform a thorough experimental analysis of the proposed formulations by measuring its generalization performance, robustness, and performance of the bounds, synthetic datasets were created. Additionally, this section has also empirically tested all formulations on protein structure data. Details of all datasets used here are following:

- **Resolution-aware Protein Structure**: A dataset based on SCOP [40] 40% sequence non-redundant dataset, taken from [16], has been used. The dataset has 15 classes (SCOP superfamilies), having 10 structures each. The experimental methodology is also similar to that used in [16], e.g. using 15 one versus all binary classifiers, where the negative data contains 10 proteins (to keep the dataset balanced) randomly chosen from all other classes. Further for each such classification problem, the whole experiment was repeated 10 times with different negative datasets.

- **Synthetic Datasets and Kernels**: 10 synthetic datasets $D_i$, $i = 1, \ldots, 10$ of various dimensions were created as follows. Each $D_i$ was generated using a Gaussian mixture of 2 Gaussian distributions. The dimension of $D_i$ was randomly chosen from $Uniform(2, 100)$. The centers for each of the 2 Gaussian were generated by selecting the
value corresponding to each dimension by randomly sampling from \( \text{Uniform}(-5, 5) \).
Each Gaussian was assumed to have a diagonal covariance matrix and each element of
the covariance matrix has been chosen from \( \text{Uniform}(0, 5) \). The center of one Gaussian
distribution was labeled as +1 and the other as −1. The Dataset \( D_i \) was then generated
by sampling \( N \) points separately from each Gaussian mixture distribution and the label
of each data point was kept the same as that of the mixture component which generated
it.
Next the construction of base kernels is described. First, a linear kernel, \( K = x_i^T x_j \),
was computed on \( D_i \), and \( L \) kernels were simulated as follows; \( K_l = K + Z_l \), where
\( Z_{lij} \) were generated using: a) Gaussian (0,1) b) Uniform [-0.5,0.5] c) centered Beta
(0.5,0.5) distributions. After that, the generated values were multiplied by a random
\( l_{ij} = l_{ji} \sim \text{Uniform}(0, 0.25|K_{ij}|) \). This creates a set of \( L \) valid positive semidefinite
kernels \( (K_l) \), and for each \( D_i \) three sets of kernels, corresponding to three distributions.
Hence forth \( D_G(S,N,L) \) will represent the set of \( S \) datasets, \( \{D_1, \ldots, D_S\} \), each dataset
having \( N \) examples per class with \( L \) kernels generated by the Gaussian distribution. Similarly
\( D_U(S,N,L) \) and \( D_B(S,N,L) \) will correspond to the Uniform and Beta distribution

**Support Information for training RSVM:** Statistical information required for \( RSVM \) has
been computed as \( a_{ij} = \min_l \left( K_{lij} - K_{ij} \right) \), \( b_{ij} = \max_l \left( K_{lij} - K_{ij} \right) \) and for Gaussian \( \beta_{ij} = \sigma^2_{ij} = \text{var}(K_{lij} - K_{ij}) \), where for other distributions initial \( \beta \) has been calculated from (5.10).

For \( RSVM_{QP} \), \( \beta \) was approximated by \( \hat{\beta} = \sqrt{\lambda_{\text{max}}(\beta \beta^T) e_{\text{max}}(\beta \beta^T) e_{\text{max}}(\beta \beta^T)^T} \), where
\( \lambda_{\text{max}}(\beta \beta^T) \) and \( e_{\text{max}}(\beta \beta^T) \) are principal eigenvalue and eigenvector of \( \beta \beta^T \). For \( RSVM_{SOCP} \),
\( \beta \) was approximated by \( \hat{\beta} = \sum_l \lambda_l(\beta) \geq \lambda_l(\beta) e_l(\beta) e_l^T(\beta) \), where \( \lambda_l(\beta) \) and \( e_l(\beta) \) are corresponding eigenvalues and eigenvectors of \( \beta \).

**Platform and package:** The three algorithms for RSVM: \( RSVM_{QP} \), \( RSVM_{SOCP} \) and \( RSVM_{QN} \),
were implemented in Matlab with the help of standard QP solver\(^3\) and Sedumi\(^4\) [33]. libSVM\(^5\)

\(^3\)http://www.mathworks.com/access/helpdesk/help/toolbox/optim/ug/quadprog.html
\(^4\)http://sedumi.ie.lehigh.edu/
\(^5\)http://www.csie.ntu.edu.tw/~cjlin/libsvm/
has been used as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.66 GHz processors and 16GB of RAM.

### 5.5.2 Comparison of Generalization Error

This section presents results which compare all the 6 proposed methods and SVM when evaluated using all three metrics described in section 5.4.2. For all the metrics, 5-fold cross-validation has been performed on 20 different datasets.

**Datasets:** The datasets $D_G(S,N,L), D_U(S,N,L), D_B(S,N,L)$ have been created with $S = 20, N = 100$ and $L = 100$, as described in Section 5.5.1. For all the metrics, 5-fold cross-validation has been performed on all the 20 datasets corresponding to each distribution. The hyperparameters ($C$ and $\epsilon$) for each classifier, were chosen using a grid search mechanism from the set $C = \{0.1, 1, 5, 10, 50, 100\}$ and $\epsilon = \{0.05 + 0.05\text{step} | \text{step} = 0, \ldots, 9\}$. For each metric, average of best cross validation accuracy, $100(1 - \text{ErrorMeasure})\%$, averaged over 20 datasets for various distributions, are reported in Table 5.1.

**Results:** Either $\text{RSVM}_{\text{SOCP}}$ or $\text{RSVM}^{(g)}_{\text{SOCP}}$ performs best in terms of all the error measures except NominalErr, clearly demonstrating the power of the proposed methods. For generating the synthetic data, in order to have a dataset where the noise is lesser than the actual kernel values, $l_{ij} = 0.25 * K_{ij}$ has been chosen. Hence, in most of the times, the $\beta$ matrix is close to PSD (few, low magnitude negative eigenvalues). As a consequence, $\text{RSVM}_{\text{SOCP}}$ or $\text{RSVM}^{(g)}_{\text{SOCP}}$ performs best. $\text{RSVM}_{\text{QN}}$ and $\text{RSVM}^{(g)}_{\text{QN}}$ follow closely, because they get stuck at local optima. $\text{RSVM}_{\text{QP}}$ and $\text{RSVM}^{(g)}_{\text{QP}}$ show intermediate performance compared to SVM.

In terms of RobustErr, SVM performs very badly, showing its lack of ability to achieve robustness. Also, $\text{RSVM}^{(g)}$ is found to perform better than $\text{RSVM}$ when the uncertainty is Gaussian. These observations are explored in detail below.

SVM in most of the times, performs best in terms of Nominal Error. This is because of the fact that NE is corresponding to $K$ and so Nominal-SVM. But it is important to note that performance of $\text{RSVM}$ is also comparable to that of SVM in terms of Nominal Error.
Table 5.1: Cross validation accuracy (in %) obtained with $\text{RSVM}^{(g)}_{\text{QP}}$, $\text{RSVM}^{(g)}_{\text{SOCP}}$, $\text{RSVM}^{(g)}_{\text{QN}}$, $\text{RSVM}_{\text{QP}}$, $\text{RSVM}_{\text{SOCP}}$, $\text{RSVM}_{\text{QN}}$, Nominal-SVM using MajorityErr (ME, (5.20)), RobustErr (RE, (5.21)), NominalErr (NE, (5.22)) and SampleErr (SE, (5.23))

<table>
<thead>
<tr>
<th></th>
<th>$\text{RSVM}^{(g)}$</th>
<th>$\text{RSVM}$</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>QP</td>
<td>SOCP</td>
<td>QN</td>
</tr>
<tr>
<td>Uniform Distribution</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>94.60</td>
<td>96.15</td>
<td>95.55</td>
</tr>
<tr>
<td>RE</td>
<td>52.60</td>
<td>93.30</td>
<td>92.85</td>
</tr>
<tr>
<td>NE</td>
<td>94.65</td>
<td><strong>96.10</strong></td>
<td>95.25</td>
</tr>
<tr>
<td>SE</td>
<td>80.30</td>
<td><strong>95.94</strong></td>
<td>95.50</td>
</tr>
<tr>
<td>Gaussian Distribution</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>95.45</td>
<td><strong>96.35</strong></td>
<td>95.95</td>
</tr>
<tr>
<td>RE</td>
<td>55.15</td>
<td><strong>93.70</strong></td>
<td>92.70</td>
</tr>
<tr>
<td>NE</td>
<td>95.05</td>
<td>95.10</td>
<td>95.55</td>
</tr>
<tr>
<td>SE</td>
<td>74.27</td>
<td><strong>95.18</strong></td>
<td>94.92</td>
</tr>
</tbody>
</table>

5.5.3 Comparison of Robustness

In the proposed RSVM (or $\text{RSVM}^{(g)}$), the effect of uncertainty during training of robust classifiers is controlled by $\varepsilon$ and hence $\kappa$ (5.16). Higher the value of $\kappa$, higher the effect of uncertainty is. For $\kappa$ equal to 0, RSVM ignores uncertainty in kernel values. This section studies behavior of all the proposed classifiers and SVM with increase of effective uncertainty. For experimental purpose following datasets were used.

Datasets: The same datasets and base kernels as in the previous subsection were used here. A consistent reduction in uncertainty for the test data points is achieved by generating them as $K_{ij}' = K_{ij} + A_{ij}'$, where $|A_{ij}'| \leq \kappa * l_{ij}$ for uncertainty with finite support, and $A_{ij}' \sim N(0, \kappa \sigma_{ij})$ for Gaussian uncertainty. Here $\kappa = \sqrt{2\log(1/\varepsilon^2)}$ and $\varepsilon \in \{0.05 + 0.05 \text{step} | \text{step} = 0, \ldots, 9\}$ with number of uncertain kernels in test data $R = 100$. The value of $C = 10$ was used in the experiment. In Figure 5.1, the RobustErr (5.21) averaged over all 20 datasets for various
Chapter 5. Handling Uncertainty in Kernel Matrices

distributions and choices of $\kappa$ have been plotted.

Figure 5.1: Robustness for RSVM, RSVM$(g)$ and Nominal-SVM using (starting from left-top clockwise) Gaussian, Uniform and $\beta(0.5,0.5)$ distribution for generating sample kernels. (Plot shows average error over 20 classifiers by fixing $C$ at 10).

**Results:** Figure 5.1 shows that, with the increase of uncertainty in test examples the RobustErr (5.21) for SVM increases more rapidly than that for RSVM$_{SOCP}$, RSVM$_{QN}$ for all types of uncertainties. This shows that non-robust classifiers, e.g. SVM, are unable to handle uncertainty, compared to the proposed robust classifiers.

RSVM$_{QP}$ performs comparably with SVM since the assumption of $\beta$ being rank 1 does not hold for the current dataset. Hence, RSVM$_{QP}$ becomes theoretically equivalent to SVM using a kernel with diagonal made heavy (see Theorem 5.5). Figure 5.2 shows that with increase in uncertainty which is IID, RSVM$_{QP}$ performs much better than Nominal-SVM in terms of
Figure 5.2: Robustness for $\text{RSVM}_{QP}$, $\text{RSVM}_{QP}^{(R)}$ and Nominal-SVM using (starting from left-top clockwise) I.I.D. Gaussian, Uniform and $\beta(0.5,0.5)$ distributions for generating sample kernels. (Plot shows average error over 20 classifiers by fixing $C$ at 10).
In Figures 5.1 and 5.2 at $\kappa = 0$, RobustErr for both $RSVM$ and $RSVM^{(g)}$ are exactly same as that of SVM. It confirms that for $\kappa = 0$, $RSVM^{(g)}$ and $RSVM$ are equivalent to SVM. Both $RSVM_{QN}$ and $RSVM^{(g)}_{QN}$ sometimes give higher error than $SOCP$ due to the solver getting stuck at local optimum.

5.5.4 Effectiveness of Bound

This section experimentally verified the effectiveness of bounds in Lemma 5.1 and Theorem 5.2, which were used to derive the RSVM formulation (5.16) from the chance constraint (5.3). For a given $\varepsilon$, $\varepsilon_{effective} = \frac{\# \{ K_i l=1\ldots L | \alpha^* Y K_l Y \alpha^* > t_i^* \} }{L}$, for the optimal values of $\alpha^*$ and $t_i^*$ solving (5.16).

Datasets: The same datasets $D_G(S, N, L)$, $D_U(S, N, L)$, $D_B(S, N, L)$ used in section 5.5.2 have been used here.

Results: Figure 5.3 plots $\varepsilon_{effective}$ vs $\varepsilon$. Ideally $\varepsilon_{effective}$ should be equal to $\varepsilon$ (shown as Ideal-case in graph). The left most plot shows that the Gaussian bound (Lemma 5.1) is much tighter than interval based bound if the uncertainty is Gaussian. Other graphs shows $\varepsilon_{effective}$ obtained from $RSVM$ for various values of $\varepsilon$, thus testing the bound in Theorem 5.2. The bound is very loose for the general distributions. However, one can observe that for the interesting range of $\varepsilon$, $\varepsilon \in [0.0.5]$, the bound is tighter than rest of the region. The bound is very tight for small values of $\varepsilon$.

5.5.5 Comparison between $RSVM$ and $RSVM^{(g)}$

Results: Figure 5.4, compares effectiveness of bound of $RSVM$ and $RSVM^{(g)}$ when the uncertainty is Gaussian. It is clear that the Gaussian bound is much tighter as expected. The left and right graph of Figure 5.5 show errors on training and test data in terms of RobustErr (5.21) respectively. $RSVM^{(g)}$ trains better when the noise is Gaussian since both training and test errors are lower.
Figure 5.3: Verification of Bound for RSVM and RSVM\(^{(g)}\) using (starting from left-top to right-bottom) Gaussian, Uniform, $\beta(0.5, 0.5)$ and $\beta(5, 5)$ distribution for generating sample kernels. (Plot shows average error over 20 classifiers).
Figure 5.4: Comparison of bound between RSVM and RSVM\(^{(g)}\) when uncertainty is Gaussian.

Figure 5.5: Comparison of RobustErr on training data (left) and on test data (Right). (Plot shows average error over 20 classifiers by fixing C at 10).
Discussion of experimental results The results on the synthetic experiments show that RSVM performs better than Nominal-SVM in terms of generalization as measured by various error measures. All the three formulations are more robust than their non-robust counterpart Nominal-SVM, while RSVM_{SOCP} and RSVM_{QN} are more robust than RSVM_{QP}. RSVM\((g)\) also performs better than RSVM while uncertainty is Gaussian. It has also demonstrated the effectiveness of bounds in Lemma 5.1 and Theorem 5.2, which were used to derive the RSVM formulation (5.16) from the chance constraint (5.3).

5.5.6 Resolution-aware Protein Structure Classification

This section presents experimental results which compare accuracy and robustness of the proposed RSVM, with state-of-the-art methods for protein structure classification.

Datasets: The dataset is described in Section 5.5.1. The coordinates of atoms in protein structures is obtained from Astral\(^6\) and their resolutions is from PDB. Let \(\mathcal{D}_{\text{Protein}} = \{(P_i, r_i, y_i)\}\) be a protein structure dataset where \(P_i\) is the \(i^{th}\) protein structure known as nominal structure and \(r_i\) is the resolution information associated to \(P_i\) and \(y_i\) is the class label. Using resolution, a set of perturbed structures \(Q_i = \{P_i^1, \ldots, P_i^L\}\) where \(P_i^l \in U(P_i)\) (1.2). This augmented dataset is used to calculate parameters for RSVM as well as other obvious extensions of SVM.

For any kernel \(K\), mean kernel \(\overline{K}_{ij} = E[K(p, p')]\) where \(K(p, p')\) is a kernel function computed between two protein structures \(p \in Q_i\) and \(p' \in Q_j\). A set of kernels consisting of \(L = 50\) uncertain kernels has been constructed. Also \(a_{ij} = \min_{p \in Q_i, p' \in Q_j} K(p, p') - E[K(p, p')]\) and \(b_{ij} = \max_{p \in Q_i, p' \in Q_j} K(p, p') - E[K(p, p')]\). For the purpose of our comparison, weighted pairwise distance substructure kernel [16] was used here. These kernel function is purely based on protein structure (specially position of \(c^a\)). Please refer to Appendix F for details.

Existing techniques: Each protein structure can be viewed as a set of perturbations of the original structure, thereby capturing the uncertainty. Hence, RSVM has been compared with

\(^6\)http://astral.berkeley.edu
Table 5.2: Table comparing \textbf{RSVM}\textsubscript{QP}, \textbf{RSVM}\textsubscript{SOCP}, \textbf{RSVM}\textsubscript{QN}, \textbf{Nominal-SVM}, \textbf{MM-M}, \textbf{MM-R} and \textbf{MI} using accuracy measures defined in section 5.4.2

<table>
<thead>
<tr>
<th>Method</th>
<th>RSVM</th>
<th>Nominal</th>
<th>MM</th>
<th>MI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>QP</td>
<td>SOCP</td>
<td>QN</td>
<td>SVM</td>
</tr>
<tr>
<td>MajorityErr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA</td>
<td>72.67</td>
<td>73.56</td>
<td>82.78</td>
<td>62.89</td>
</tr>
<tr>
<td>F1</td>
<td>73.49</td>
<td>74.35</td>
<td>82.95</td>
<td>63.50</td>
</tr>
<tr>
<td>RobustErr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA</td>
<td>27.11</td>
<td>50.33</td>
<td>66.44</td>
<td>34.56</td>
</tr>
<tr>
<td>F1</td>
<td>26.81</td>
<td>50.28</td>
<td>66.36</td>
<td>34.07</td>
</tr>
<tr>
<td>NominalErr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA</td>
<td>66.22</td>
<td>66.11</td>
<td>75.44</td>
<td>75.89</td>
</tr>
<tr>
<td>F1</td>
<td>65.52</td>
<td>66.41</td>
<td>78.44</td>
<td>76.12</td>
</tr>
<tr>
<td>SampleErr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TA</td>
<td>66.50</td>
<td>66.65</td>
<td>76.00</td>
<td>61.02</td>
</tr>
<tr>
<td>F1</td>
<td>65.13</td>
<td>65.16</td>
<td>75.80</td>
<td>60.86</td>
</tr>
</tbody>
</table>

The normalized set multi-instance kernel (\textbf{MI}) \cite{41}. For a given kernel $K$, normalized set kernel is defined as $K_{\text{multi}}(P_i, P_j) = \frac{\sum_{p \in Q_i, p' \in Q_j} K(p, p')}{\sqrt{\left(\sum_{p \in Q_i} K(p, p')\right)\left(\sum_{p' \in Q_j} K(p, p')\right)}}$.

Another option is to interpret every perturbation of a protein structure as an independent data point. Hence, comparison of the \textbf{RSVM} was done with (\textbf{MM-R})(\textbf{SVM} considering each perturbed structure as individual data points) and (\textbf{MM-M}) (\textbf{SVM} considering $\overline{K}$ as kernel). The Nominal-SVM (\textbf{SVM} with kernel based on protein structure reported in \textit{PDB files}) is used for benchmarking the result.

\textbf{Results:} Table 5.2 reports results for \textbf{RSVM} and state of the art methods using both standard and robust error measures defined in Section 5.4.2 by performing LOO cross validation. Hyper-parameters (C and/or $\varepsilon$) for each classifier were tuned separately using a grid search. Total Accuracy (TA) and F1 score have been reported, since the problem is posed as a one-vs-all classification scheme, which can also be thought as retrieval of the positive class. All reported results are mean over 10 different datasets, where negative dataset were selected randomly. Also, note that for SVM with MI kernel, one label is given to every set following \cite{41} method and it was reported as the majority error for MI kernel.

It is clear that $\textbf{RSVM}_{QN}$ performs significantly better than rest of the methods, both in terms
of Accuracy (measured by MajErr) and Robustness (measured by RobustErr). This indicates that use of resolution information improves the overall classification accuracy. Moreover, very low values of accuracy corresponding to RobustErr for SVM and other competing methods suggests that the SVM classification is not robust to perturbations in coordinates of atoms within the resolution. The fact that other RSVM formulations perform worse than $RSVM_{QN}$ indicates that assumptions used to derive other formulations, e.g. Rank 1 or PSD, do not hold for this dataset.

In terms of RobustErr, $RSVM_{QP}$ performs worse than $RSVM_{SOCP}$, confirming the fact that PSD assumption is more appropriate for robustness than rank one assumption for this dataset. The simple heuristic of using all the perturbed samples (MM-R) performs very well in terms of robustness, which is intuitive. However, the computational complexity of this method is $O(L^2)$ higher than other methods, which can be prohibitive for many cases.

### 5.6 Summary

This chapter has presented an optimization problem (5.16), which is robust to uncertainty in the kernel matrix. The formulation applies to Gaussian uncertainty as well as to arbitrary distributions with finite support. For the finite support case the formulation is derived from a novel large deviation inequality, stated in Theorem 5.2. The large deviation inequality is of independent interest and applies more generally to problems involving traces of random matrices. An interesting result is, for i.i.d uncertainty the formulation reduces to SVM (Theorem 5.5). Solving co-positive programs are extremely difficult but for positive semidefinite $\beta$ the formulation is second-order-cone representable and can be solved by SOCP. The formulation when applied to protein structure classification yields significantly improved results.
Chapter 6

Affine Uncertainty set for handling Uncertainty in Kernel Matrices: Robust Optimization Approach

Abstract

This chapter considers the problem of designing classifiers when the kernel matrix is uncertain. The previous chapter studied a chance-constraint based setup when the uncertainty in the kernel matrix is modeled as independent noise over the kernel entries. The independence assumption is restrictive and mayn’t always define a valid kernel. To alleviate this problem, an affine set based alternative is proposed. Using a robust optimization framework the resultant problem is posed as a minimax problem. This formulation here has been applied on the same problem as described in the previous chapter. Since this formulation models uncertainty in kernels in a better and valid way, it achieves better generalization.

6.1 Introduction

Given a dataset $\mathcal{D} = \{(x_i, y_i)|i = 1, \ldots, n\}$ the SVM dual formulation [1] can be written as:

$$\max_{\alpha \in \mathbb{R}^n, t} \quad \alpha^\top e - \frac{1}{2} t \quad \text{s.t.} \quad \alpha^\top Y K Y \alpha \leq t$$  \hspace{1cm} (6.1)
where $S_n = \{\alpha|0 \leq \alpha_i \leq C, \sum_{i=1}^{n} \alpha_i y_i = 0\}$ and $Y = \text{diag}(y_i)$. The kernel matrix, $K$, is a $n \times n$ matrix, where each entry of the matrix $K$, is defined by $K_{ij} = K(x_i, x_j)$. The kernel function, $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, plays the role of dot product between implicitly defined feature map over examples $x_i, x_j \in \mathcal{X}$. Hence the matrix $K$ needs to be positive semi-definite (see e.g. [42]). Kernel based classifiers are extremely powerful as they can define complicated decision boundaries as following:

$$f(x) = \text{sign} \left( \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) \right).$$  \hspace{1cm} (6.2)

Importance of designing a robust classifier when the kernel matrix $K$ is uncertain has been already discussed in the previous chapter along with the main motivating force behind it. This chapter is going to deal with the same problem again. But unlike previous chapter, this chapter considered the fact that all uncertain kernel matrix is also a valid kernel matrix.

### 6.1.1 Review of Past Work

The previous chapter studied this problem where the uncertainty is characterized by independent noise in each of the entries of the kernel matrix. More specifically let $\overline{K}$ be a given kernel matrix, such that it is symmetric and positive semidefinite. The chance constraint program is used as a surrogate to the following problem (stated in 5.3).

$$p^* = \max_{t, \alpha \in S_n} \alpha^\top e - \frac{1}{2} t$$  \hspace{1cm} (6.3)

$$\text{s.t.} \quad \text{Prob}(\alpha^\top Y(\overline{K} + Z)Y \alpha \leq t) \geq 1 - \varepsilon $$  \hspace{1cm} (6.4)

where $\varepsilon < 0.5$, and entries of $Z$ are independently distributed. It is further assumed that the distribution could be Gaussian or has only finite support. Then the final approximated problem (5.16) has been solved. But this suffers from two drawbacks. First unless the matrix $\beta$ is psd, problem (5.16) is not convex. Second the matrix, $\overline{K} + Z$, need not be psd, and hence can’t model a valid kernel matrix. Indeed for a general $\beta$ the problem is non-convex and a locally
optimal Quasi newton procedure has been proposed for solving this problem.

6.1.2 Contribution

The approach outlined in this chapter overcomes both these drawbacks. It employs a Robust Optimization (RO) approach where instead of the probabilistic description of the uncertainty, a geometric description is used. The uncertainty in the kernel matrix is modeled by a convex and bounded uncertainty set which encompasses possible realizations of $K$. The realizations are modeled by an affine combination of a finite set of fixed kernels. This new approach leads to an convex conic optimization problem which could be cast as SOCP.

6.1.3 Structure of the Chapter

The RO approach for designing robust SVMs is discussed in Section 6.2. In Section 6.3 the ground for a comprehensive computational study by introducing various prediction rules and related error metrics is prepared. The results of the computational study are described in Section 6.4.

6.2 Affine Uncertainty set for Handling Uncertainty in Kernel Matrices

This section introduces an uncertainty set of psd matrices, which is affine in nature and studies the resultant robust SVM problem using an RO approach.

Consider an uncertain optimization problem where $f, g_i : \mathcal{P} \subset (\mathbb{R}^n \times \mathbb{R}^k) \Rightarrow \mathbb{R}$

\[
\begin{aligned}
\text{(UOP)} & \quad \min_{x \in \mathbb{R}^n} f(x, \Psi) \\
& \quad g_i(x, \Psi) \leq 0 \quad i = 1, \ldots, m
\end{aligned}
\]  

(6.5)

where $\Psi \in \mathbb{R}^k$ is a vector of uncertain parameters. The (UOP) is in fact a family of problems -one for each realization of $\Psi$. In the RO framework the information related to $\Psi$ is modeled
as a geometric uncertainty set \( \mathcal{U} \subset \mathbb{R}^k \) and the family of problems, (UOP) is replaced by its robust counterpart (RC):

\[
\text{(RC)} \quad \text{sol}^* = \min_x \max_{\Psi \in \mathcal{U}} f(x, \Psi) \\
g_i(x, \Psi) \leq 0 \quad \forall \Psi \in \mathcal{U} \quad i = 1, \ldots, m 
\]

(6.6)

A solution of (RC) is guaranteed to be feasible to (UOP) for any realization of \( \Psi \in \mathcal{U} \) and the objective function is no worse than \( \text{sol}^* \).

The uncertainty set \( \mathcal{U} \) is typically a polytopes or ellipsoid or intersection of such sets. These sets yield useful models of uncertainty, which lead to tractable optimization problems [23]. A general representation of \( \mathcal{U} \) is as follows

\[
\mathcal{U} = \{ \Psi = \bar{\Psi} + \sum_{i=1}^{L} \eta_i \Psi^i ||\eta|| \leq \rho \}
\]

where \( \bar{\Psi} \) is the nominal value of the uncertain vector \( \Psi \), the vectors \( \Psi^i \) are possible scenarios of it, and \( \eta \) is a perturbation vector. The norm is suitably defined to capture the geometry of the set. As an example, Consider the ellipsoidal set

\[
\mathcal{U}_{\text{ellipsoid}} = \{ \Psi | (\Psi - \bar{\Psi})^\top Q (\Psi - \bar{\Psi}) \leq \rho \}
\]

where \( Q \in \mathcal{S}_+^n \) and is positive definite. It is easily seen that the set can be represented by \( \mathcal{U} \) with \( \Psi^i \) being the columns of \( Q^{-\frac{1}{2}} \) and where \( ||\cdot|| \) is the \( L_2 \) norm.

In general the (RC) of an (UOP) may have infinite number of constraints and is often NP hard. However in several important cases it reduces to a polynomially solvable convex optimization problem. Unfortunately a detailed discussion is beyond the scope of this chapter and please see [23] for a comprehensive treatment of RO problems.
6.2.1 Modeling uncertainty sets for Kernel matrices

The setup for the problem is as follows. There is a kernel function $K$ which when presented with a pair of observations $z, z' \in \mathcal{X}$ computes $K(z, z')$. We assume that we do not have direct access to this value, instead we have access to a close approximation, $\overline{K}(z, z')$, which is again a kernel function evaluated at $(z, z')$. The function $\overline{K}(\cdot, \cdot)$, will be called the nominal kernel and when evaluated at any $z, z'$ the value will be referred to as the nominal value. The difference between actual value and the nominal value is expressed by a linear combination of known $L$ kernel functions, $K_l, l = 1, \ldots, L$ evaluated at points $z, z'$, as follows:

$$K(z, z') - \overline{K}(z, z') = \sum_{l=1}^{L} \eta_l K_l(z, z')$$

When there is no uncertainty $K(z, z') = \overline{K}(z, z')$ and $\eta = 0$. The value of $K(z, z')$ lies in the uncertainty set

$$\{\overline{K}(z, z') + \sum_{l=1}^{L} \eta_l K_l(z, z') | \| \eta \|_p \leq \kappa, \eta_l \geq 0 \forall l = 1, \ldots, L\} \quad (6.7)$$

where $\| \eta \|_p, p \geq 1$ denotes the $l_p$ norm on $\eta$. The constraint $\eta_l \geq 0$ is needed to ensure that each element in the set represents a valid kernel evaluation. The quantity $\kappa$ measures the quality of approximation and hence the uncertainty. If $\kappa = 0$ then we have no uncertainty. As $\kappa$ increases the uncertainty set increases. In the sequel we will refer to $\overline{K}, K_l$ as base kernels.

We impose the uncertainty set (6.7) to all examples of interest. The described uncertainty set immediately leads to the following model of uncertainty on the kernel matrix corresponding to the training set.

$$\mathcal{U}(\kappa) = \{K = \overline{K} + \sum_{l=1}^{L} \eta_l K_l, \| \eta \|_p \leq \kappa, \eta_l \geq 0, l = 1, \ldots, L\} \quad (6.8)$$

The matrices $\overline{K}, K_l \in \mathcal{S}_{n^+}$ are obtained by evaluating the known kernel functions $\overline{K}, K_l$ on the training set. As any $K \in \mathcal{U}(\kappa)$ is always positive semi-definite, the set $\mathcal{U}(\kappa)$ defines a valid model for describing uncertainty in psd matrices. In a later subsection we will discuss the
relevance of this setup to Protein structure classification problem.

The Robust SVM problem (6.1) with uncertain $K$, as characterized in (6.8), can now be cast as follows

$$\max_{\alpha \in S_n} \min_{K \in \mathcal{K}} \frac{1}{2} \alpha^T Y K Y \alpha + \alpha^T e$$

or more explicitly

$$\max_{\alpha \in S_n} \min_{\eta \in \mathcal{R}_p(K)} \frac{1}{2} \alpha^T Y K Y \alpha - \frac{1}{2} \alpha^T Y \sum_{l=1}^{L} (\eta_l K_l) Y \alpha + \alpha^T e \quad (6.9)$$

where

$$\mathcal{R}_p(K) = \{ \eta \| \eta \|_p \leq \kappa \} \quad (6.10)$$

Note that in the latter problem the constraint $\eta \geq 0$ is dropped. Indeed if we define $a_l = \alpha^T Y K_l Y \alpha$ then $a_l \geq 0$ as $K_l \in \mathcal{K}_n^+$. The optimal $\eta$ is the solution of $\max_{\eta \in \mathcal{R}_p} a^T \eta$, which occurs at $\eta_l = \kappa \frac{a_l}{\| a \|_p} \geq 0$ where $\| \cdot \|_p$ is the dual norm of $\| \cdot \|_p$, with $q = \frac{p}{p-1}$ for any $p > 1$. For $p = 1$ one needs to observe that optimality is achieved at $\eta_l \geq 0$. Indeed there exists some $l$ such that $a_l = \| a \|_\infty$. The optimal $\eta$ is given by the condition that $\sum_l a_l = \| a \|_\infty \eta_l = 1$ and $\eta_l \geq 0$. If $a_l < \| a \|_\infty$ then $\eta_l$ is strictly 0. At optimality $\eta_l \geq 0$, and hence the resultant optimal kernel lies in $\mathcal{K}(\kappa)$. For the special case $p = 2$, the problem (6.9) can be stated as a Second Order Cone Program (SOCP)\(^1\) as follows:

$$(\text{USSVM}) \quad \min_{\alpha \in S_n, \eta, \tau} \frac{1}{2} \alpha^T Y K Y \alpha + \frac{1}{2} \alpha^T e$$

s.t. $\| a \|_2 \leq \tau$

$$\alpha^T Y K_l Y \alpha \leq a_l \forall l = 1, \ldots, L \quad (6.11)$$

SOCP problems such as (6.11) can be solved by Interior Point (IP) algorithms (e.g., CPLEX, MOSEK, Sedumi) and will be denoted by Uncertainty-Set SVM (USSVM)

---

\(^1\)This is also true for any ($p > 1$) because any $p$-norm can be represented by conic quadratic inequalities. For a more detailed discussion on this issue see [19]
Generic IP solvers are inadequate for large scale classification problems. Instead, we demonstrate here that the minimax reformulation, (6.9), admits algorithms which are better suited to large scale problems. For simplicity of exposition we will consider the case \( p = 2 \). The results can be easily extended for the general case, \( p > 1 \). Before we discuss the algorithmic aspects it maybe useful to discuss the computation of the base kernels.

### 6.2.2 Evaluation of base kernels

Recall that in the protein structure classification problem each observation is specified by a \((P, U(P), y)\), where \(P\) (see (1.1)) is the nominal structure, \(U(P)\) (see (1.2)) is the uncertainty set specified by the resolution and \(y\) is the label. We discuss this problem in a formal setup and motivate the uncertainty set described in (6.7).

To closely parallel the protein structure classification setting we consider the following setup. We are given a dataset \( D = \{ (\bar{x}_i, U_i) | x_i \in U_i, U_i \in \mathcal{X}, i = 1, \ldots, m \} \) where an observation, \( x_i \), is not directly specified, instead a nominal value, \( x_i \), and an uncertainty set \( U_i \) are given. When there is no uncertainty the set \( U_i \) reduces to only \( x_i \). The actual observation \( x_i \) is a member of \( U_i \). We are also given a kernel function \( K: \mathcal{X} \times \mathcal{X} \to \mathbb{R} \). We make no assumptions about the functional form of \( K(\cdot, \cdot) \), and we assume that when presented with any pair \( z, z' \in \mathcal{X} \) it returns a value \( K(z, z') \).

Let us now consider the computation of \( K(x_i, x_j) \). Since \( x_i, x_j \) are uncertain the precise value of \( K(x_i, x_j) \) but it for sure lies in the set

\[
\mathcal{U}(x, x') = \{ K(z, z') | z \in U(x), z' \in U(x') \}
\]

If there is no uncertainty then the set \( \mathcal{U}(x, x') \) is a singleton, \( K(\bar{x}, \bar{x}') \). In the presence of the uncertainty the elements of \( \mathcal{U} \) lie in the closed interval \([\min_{z \in U(x), z' \in U(x')} K(z, z'), \max_{z \in U(x), z' \in U(x')} K(z, z')\] \). However as the functional form of \( K \) is not known, computation of the extrema of \( \mathcal{U} \) is intractable. Clearly to make progress we will need to build an alternate description of the set \( \mathcal{U} \). To this end we resort to a sampling procedure. We make \( L \) independent draws from the uncertainty sets. In the \( l \)th draw we obtain \( O^l = \{ z_1^l, \ldots, z_m^l \} \) where \( z_i^l \) is an independent and
uniform draw from $U_i$. For a given $O^i$ we invoke the kernel function $K$ to obtain an $m \times m$ kernel matrix $K_i$ where $K_i(x_i, x_j) = K(z_i^l, z_j^l)$. Once $K_i$ are determined we propose to approximate $\mathcal{U}$ by uncertainty set described in (6.7).

### 6.3 Prediction rules and Error metrics

The problem of attaching a label on an uncertain point associated with uncertain kernel and evaluating the prediction rule has already been discussed in the previous chapter (see Section 5.4). But the prediction rule discussed in Section 5.4.1 is not directly applicable to the case of affine set uncertainty. In this section, a similar prediction rule is described and also applicability of all the error measures described in Section 5.4.2 to this prediction rule is discussed.

**Choice of bias** Before going forward, the choice of bias of the classifier needs to be clarified here. For a given $K, K_1, \ldots, K_L$ let the optimal solution of (6.9) be $\alpha^*$ and $\eta^*$. This $\eta^*$ can be viewed as defining an effective kernel

$$K = \overline{K} + \sum_{l=1}^{L} \eta^l K_l$$

and hence the bias can be computed as

$$bias = \frac{1}{\# SV} \sum_{j \in SV} [y_j - \sum_i y_i \alpha_i^* K_{ij}]$$

where $SV$ is an index set of support vectors.

#### 6.3.1 Prediction Rules

Given a test data-point $x_t$ and support vector $x_i$ the kernel function is assumed to be uncertain but it is known that $L$ different values defined as in $K_l(x_t, x_i) \forall l = 1, \ldots, L$ are available. Let $\{\eta^{tst}_1, \ldots, \eta^{tst}_R\}$, be $R$ random instances of $\eta^{tst}$ where each $\eta^{tst} \in \mathcal{B}_p(\kappa)$ (see (6.10)). Each
choice of $h^{\text{tst}}$ generates a realization of kernel

$$K_{p}^{\text{tst}}(x_t, x_i) = \bar{K}(x_t, x_i) + \sum_{l=1}^{L} \eta_l^{\text{tst}} K_l(x_t, x_i)$$

where $\bar{K}(x_t, x_i) = \frac{1}{L} \sum_{l=1}^{L} K_l(x_t, x_i)$. This choice of $\eta^{\text{tst}}$ makes it consistent with the model of uncertainty as defined in (6.8). Similar to (5.19), the final prediction is obtained by taking a majority vote on individual predictions as follows

$$y_t^{pr} = \text{sign} \left( \sum_{r=1}^{R} y_r^t \right), \quad y_r^t = \text{sign} \left( \sum_i \alpha_i y_i K_{r}^{\text{tst}}(x_t, x_i) + \text{bias} \right)$$

(6.12)

where

$$K_{r}^{\text{tst}}(x_t, x_i) = \bar{K}(x_t, x_i) + \sum_{l=1}^{L} \eta_l^{\text{tst}} K_l(x_t, x_i).$$

(6.13)

### 6.3.2 Error Metrics

Given a test dataset $D_{\text{test}} = \{(x_t, y_t) | t = 1, \ldots, n_{\text{tst}}\}$ where for each test-point there are multiple instances of the kernel values. One can use (6.12) for prediction and hence use all the error measures defined in Section 5.4.2 to evaluate the performance of USSVM.

### 6.4 Experimental Evaluation

This section presents experimental evaluation of the formulations listed in Table 6.1\(^2\). In particular it would be interesting to explore the following questions.

1. Comparison of USSVM against the non-robust Nominal SVM

2. Robustness of the USSVM and RSVM formulation when faced with an uncertainty of type described in $\mathcal{U}(\kappa)$.

Experimental results extensively compare USSVM, RSVM with the Nominal-SVM. The nominal SVM is defined as the traditional SVM (6.1) with a known kernel matrix $\bar{K}$. It would

\(^2\)Relevant data and scripts are available at [http://mllab.csa.iisc.ernet.in/~sahely/uncertainkernel.html](http://mllab.csa.iisc.ernet.in/~sahely/uncertainkernel.html)
Table 6.1: Various formulations and associated information required.

<table>
<thead>
<tr>
<th>Formulation</th>
<th>Required information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal – SVM (6.1)</td>
<td>$K$</td>
</tr>
<tr>
<td>RSVM (5.16)</td>
<td>$K$ and $a_{ij}, b_{ij}$</td>
</tr>
<tr>
<td>USSVM (6.11)</td>
<td>$K$ and $K = {K_1, \ldots, K_L}$ (set of valid kernels)</td>
</tr>
</tbody>
</table>

be interesting to see how Nominal-SVM performs in comparison with the proposed robust formulations. A brief summary of the formulations which are compared is presented in Table 6.1.

The section is organized as follows. It begins with a brief description of datasets in Section 6.4.1. The first issue is discussed in Section 6.4.2. A comparative study of robustness is presented in Section 6.4.3.

### 6.4.1 Datasets

The synthetic datasets were built to test the generalization and robustness properties of the proposed formulations. In addition, all of them have been empirically tested on protein structure data as described below.

- **Resolution-aware Protein Structure:** The same dataset described in Section 5.5.1 is used here. The dataset has 15 classes (SCOP superfamilies), having 10 structures each. 15 binary classification problems are posed for identifying each class, where the negative class contained 10 proteins (to keep the dataset balanced) randomly chosen from all other classes. Furthermore, for each such classification problem, the procedure was repeated 10 times, giving rise to 10 different datasets, where the positive class remained constant, but the negative class varied.

- **Synthetic Datasets and Kernels:** 10 synthetic datasets were created $D_i$, $i = 1, \ldots, 10$ of various dimensions as follows. Each $D_i$ was generated using a Gaussian mixture of 4 Gaussian distributions. The dimension of $D_i$ was randomly chosen from $\text{Uniform}(2, 100)$. The centers for each of the 4 Gaussian were generated by selecting the value corresponding to each dimension by randomly sampling from $\text{Uniform}(-5, 5)$. Each Gaussian was
assumed to have a diagonal covariance matrix and each element of the covariance matrix has been chosen from $\text{Uniform}(0, 5)$. At first, labels were assigned to the centers of each Gaussian distribution according to $\text{sign}(w^\top x)$, where $w$ is a random vector with $\|w\| = 1$. The dataset $D_i$ was then generated by sampling $N$ points separately from the Gaussian mixture distribution consisting only positively labeled mixture components and then by sampling another $N$ points from the Gaussian mixture distribution consisting only negatively labeled mixture components. The label of each data point was kept same as that of the mixture component which generated it.

Next paragraph describes the construction of base kernels. First, a linear kernel, $K_{ij} = x_i^\top x_j$, was computed on $D_i$, and $L$ kernels were simulated as follows; $K_l = \overline{K} + Z_l Z_l^\top$, where $Z_{ij}$ were generated using: a) Gaussian $(0,1)$ b) Uniform $[-1,1]$ c) centered Beta $(0.5,0.5)$ distributions. After that, the generated values were multiplied by a random $l_{ij} \sim \text{Uniform}(0, 0.05|\overline{K}_{ij}|)$. This creates a set of $L$ valid positive semidefinite kernels ($K_l$), and for each $D_i$ three sets of kernels, corresponding to three distributions.

$\mathbf{DK}_G(S, N, L)$ denotes the set of $S$ datasets, $\{D_1, \ldots, D_S\}$, each dataset having $N$ examples per class with $L$ kernels generated by the Gaussian distribution. Similarly $\mathbf{DK}_U(S, N, L)$ and $\mathbf{DK}_B(S, N, L)$ will correspond to the Uniform and Beta distribution respectively.

**Support Information for training RSVM** In the setting of the paper a set of kernel matrices $\mathcal{K} = \{K_1, K_2, \ldots, K_L\}$ are specified. The formulation RSVM needs support information (Please see Table 6.1.) which could be extracted as follows

$$K = \frac{1}{L} \sum_{l=1}^{L} K_l \quad a_{ij} = \min_l (K_{ij}) \quad b_{ij} = \max_l (K_{ij})$$

**Platform and package** The algorithms USSVM and RSVM have been implemented in Matlab with the help of Sedumi$^3$ [33]. libSVM$^4$ has been used as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.66 GHz processors and

$^3$http://sedumi.ie.lehigh.edu/

$^4$http://www.csie.ntu.edu.tw/ cjlin/libsvm/
Chapter 6. Affine Uncertainty set of Uncertain Kernel

16GB of RAM.

### 6.4.2 Comparison of Generalization Error

This section have performed a thorough experimental analysis of the proposed formulations measuring their generalization performances. The proposed formulation USSVM has been compared here with the RSVM (5.16), Nominal – SVM using the four metrics described in Section 5.4.2.

**Datasets and Kernels:** Datasets $\mathbf{DK}_G(S,N,L)$, $\mathbf{DK}_U(S,N,L)$, $\mathbf{DK}_B(S,N,L)$ were created with $S = 10, N = 250, L = 200$, as described in Section 6.4.1. The uncertain kernels were generated by using $R = 100$ and $\kappa = 1$ (see Section 6.3). For each metric, the cross-validation accuracies, $100(1 - \text{ErrorMeasure})\%$ were averaged over 10 datasets for various distributions and have been reported in Table 6.2. The hyper-parameters ($C$ and $\varepsilon$) for each classifier, were chosen using a grid search mechanism from the set $C = \{0.1, 1, 5, 10, 50, 100, 200, 500\}$ and $\varepsilon = \{0.05 + 0.05\times\text{step}|\text{step} = 0, \ldots, 9\}$. Note that, $\mathbf{DK}_G$ refers to the Gaussian case, $\mathbf{DK}_U$ refers to Uniform case and $\mathbf{DK}_B$ refers to the Beta case.

**Results:** It is clear that, for all the datasets USSVM beats RSVM in terms of Robust Error. It indicates the fact that, USSVM is more robust to uncertain kernels than that of RSVM. However, for the other error measures, performance of USSVM is very similar to that of SVM. In terms of the RobustErr, SVM performs very badly for all kind of uncertainties. This observation shows SVM’s lack of ability to achieve robustness. These observations are explored in detail below. SVM in most of the time performs best in terms of Nominal Error. This is because of the fact that NomErr is corresponding to $\overline{K}$ and so is Nominal-SVM. But it is important to note that performance of USSVM is also comparable to that of SVM in terms of Nominal Error.
Table 6.2: Cross-validation accuracy (%) obtained with USSVM, RSVM and Nominal SVM using RobustErr (RE, (5.21)), MajorityErr (ME, (5.20)), NominalErr (NE, (5.22)) and SampleErr (SE, (5.23)). All values reported here are 100(1 – Errormeasure)%

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>USSVM</th>
<th>RSVMQP</th>
<th>RSVM_{SOCP}</th>
<th>RSVM_{QN}</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DK_G</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>89.10</td>
<td>85.00</td>
<td>89.00</td>
<td><strong>89.20</strong></td>
<td>78.70</td>
</tr>
<tr>
<td>RE</td>
<td><strong>75.60</strong></td>
<td>35.90</td>
<td>66.90</td>
<td>57.20</td>
<td>26.10</td>
</tr>
<tr>
<td>NE</td>
<td>89.00</td>
<td>84.70</td>
<td>89.00</td>
<td>88.90</td>
<td><strong>90.90</strong></td>
</tr>
<tr>
<td>SE</td>
<td><strong>88.94</strong></td>
<td>80.19</td>
<td>88.81</td>
<td>88.82</td>
<td>77.46</td>
</tr>
</tbody>
</table>

| **DK_U** |       |        |             |           |     |
| ME       | **91.00** | 88.00  | 89.00       | 89.00     | 87.90 |
| RE       | **85.40** | 62.60  | 78.40       | 73.40     | 48.80 |
| NE       | 89.90 | 90.00  | 90.00       | 89.10     | **90.90** |
| SE       | **88.98** | 87.97  | 88.89       | 88.61     | 87.97 |

| **DK_B** |       |        |             |           |     |
| ME       | **88.00** | 86.70  | 87.00       | 87.10     | 80.90 |
| RE       | **85.90** | 72.80  | 82.20       | 82.80     | 62.00 |
| NE       | 89.80 | 91.00  | 91.00       | **91.10** | 90.90 |
| SE       | **86.99** | 85.31  | 86.87       | 86.89     | 82.01 |

### 6.4.3 Comparison of Robustness

In the proposed USSVM, the effect of uncertainty in training data is controlled by $\kappa$ (6.9). Higher the value of $\kappa$ is, higher is the effect of uncertainty. For $\kappa = 0$, USSVM ignores uncertainty in the kernel values.

**Datasets:** The same datasets and base kernels described in the previous subsection have been used here, and uncertainty is taken from $\kappa \in \{0.1, 0.2, 0.3, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 5\}$ with $R = 100$. In Figure 6.1, the RobustErr (5.21) averaged over all 10 datasets for various distributions and choices of $\kappa$ have been reported. The value of $C = 100$ was used. Again $\text{DK}_G$ will refer to the Gaussian distribution, $\text{DK}_U$ refers to the uniform case, and $\text{DK}_B$ refers to the Beta distribution.

**Results:** In Figure 6.1 at $\kappa = 0$, the RobustErr for USSVM is exactly same as that of SVM. It confirms the fact that at $\kappa = 0$, USSVM is equivalent to SVM, as there is no uncertainty. Figure 6.1 shows that, with the increase of uncertainty in the test examples, the RobustErr (5.21) for
Figure 6.1: Robustness of \textbf{USSVM} and \textbf{Nominal-SVM} on (starting from left-top clockwise) \( \text{DK}_G; \text{DK}_U; \text{DK}_B \). (Plot shows average error 100(\textit{ErrorMeasure})\% over 10 classifiers by fixing C at 100).

SVM increases more rapidly than that for \textit{USSVM} for all the 3 datasets. This shows that, non-robust classifiers, e.g. SVM, are unable to handle uncertainty compared to the proposed robust classifiers.

\textbf{Discussion of experimental results}  The results on the synthetic experiments show that \textbf{USSVM} performs better than \textbf{RSVM} in terms of generalization as measured by various error measures. Proposed formulation \textbf{USSVM} is more robust than the non-robust Nominal SVM.
6.4.4 Resolution-aware Protein Structure Classification

Current section experimentally explores the performance of the proposed formulations on a real world problem of protein structure classification. As briefly mentioned in the Chapter 1, protein structure information is uncertain and subject to resolution information, which is usually ignored in the literature. [43] and Chapter 5 use the chance constraint approach to pioneer the use of resolution information for classifying protein structures. The Affine Uncertainty model was applied to this problem and it is observed that appropriate utilization of resolution information does make classifiers much more robust. This section presents experimental results in order to compare the performance of the proposed USSVM, with RSVM and other state of the art methods for protein structure classification.

Datasets and Experimental Methodology: The dataset is described in Section 6.4.1. The experimental methodology is also similar to that used in [16] and [43]. Leave-One-Out (LOO) cross validation using SVM was performed on all 15 of such classification problems. Recall that for each classification problem, 10 different datasets were generated by sampling the negative class and the reported results are averaged over all the 10 different datasets. In all cases the reported accuracy is computed as 100\((1 – ErrorMeasure)\)%.

The Base Kernels: Recall that in the protein structure classification problem each observation is specified by a \((P, U(P), y)\), where \(P\) (see (1.1)) is the nominal structure, \(U(P)\) (see (1.2)) is the uncertainty set specified by the resolution and \(y\) is the label. We discuss this problem in a formal setup and motivate the uncertainty set described in (6.7).

One can create a set of uncertain kernels, where \(K(p, p')\) is a kernel function computed between two protein structures \(p \in U(P_i)\) and \(p' \in U(P_j)\). For experiments, a set of kernels consisting of \(L = 50\) base kernels were generated. Denoting the kernel matrices by \(\{K_1, \ldots, K_L\}\), the uncertainty set is defined as \(\mathcal{W}(1)\) (see (6.8)) with \(K = \frac{1}{L} \Sigma_{l=1}^{L} K_l\) and \(\kappa = 1\). Given the base kernels, the prediction process is same as reported in Section 6.3 with \(R = 100\) and \(\kappa = 1\). For the purpose of comparison, weighted pair-wise distance substructure kernel [16] was used. These kernels are purely based on protein structure (specially position of \(c^a\)). Please refer to
Appendix F for details.

For R SVM, mean kernel $K_{ij} = E[K(p, p')]$, and

$$a_{ij} = \min_{p \in U(P_i), p' \in U(P_j)} K(p, p')$$

$$b_{ij} = \max_{p \in U(P_i), p' \in U(P_j)} K(p, p')$$

The next paragraph has described experimental results on a real life resolution-aware protein structure dataset.

**Results:** Table 6.3 reports results for R SVM, USSVM and SVM$_{PDB}$ (SVM with kernel based on protein structure reported in PDB files) and Nominal – SVM (SVM considering $K$ as kernel) using both standard and robust error measures defined in Section 5.4.2 in the Leave-One-Out (LOO) procedure. Hyper-parameters (C and/or $\varepsilon$) for R SVM and C for USSVM and SVM’s were tuned separately using the grid search mechanism. As this is a 15 class classification problem and a “one-vs-all” setting was followed, Positive Accuracy (PA) [percentage of positive text data points classified correctly], Negative Accuracy (NA) [percentage of negative text data points classified correctly] and Total Accuracy (TA) [percentage of total text data points classified correctly] were reported here. All the reported results are averaged over 10 different datasets, where negative dataset were selected randomly. As observed in Table 5.2 of the previous chapter (Chapter 5) that the behavior of all formulation with respect to F-score is similar to that with respect to Total Accuracy, F-scores are not reported here.

It is clear that R SVM, and USSVM perform significantly better than their non-robust counterparts, both in terms of Accuracy (measured by MajorityErr) and Robustness (measured by RobustErr). This result indicates that, the use of resolution information improves the overall classification accuracy. Moreover, very low values of accuracy corresponding to RobustErr for SVM and other competitive methods suggests that the SVM classification is not robust to perturbations in coordinates of atoms even within the resolution. In fact, USSVM even beats R SVM in terms of Robustness, because, unlike R SVM, USSVM formulation considers only the valid kernels without any other assumption.
Table 6.3: Comparison of USSVM, RSVM, Nominal-SVM and SVM\textsubscript{PDB} using accuracy measures, 100(1 – Error\textit{Measure})\%, where Error measures are defined in Section 5.4.2

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>USSVM</th>
<th>RSVM</th>
<th>Nominal-SVM</th>
<th>SVM\textsubscript{PDB}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OP</td>
<td>SOCP</td>
<td>QN</td>
<td>PA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>64.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>64.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>76.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>60.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>48.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>41.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>43.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>60.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>48.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>41.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>43.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>33.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>44.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>37.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>26.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12.13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28.86</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>17.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>19.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>43.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>58.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>56.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>59.47</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>44.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>46.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>45.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>47.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>60.26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.79</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.57</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>70.47</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>61.30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40.64</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40.98</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>39.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>41.85</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>55.75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.72</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.77</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>56.58</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50.97</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.57</td>
</tr>
</tbody>
</table>

6.5 Summary

In this chapter the problem of designing classifiers with uncertain kernel matrix is studied again. Unlike previous chapter, the affine uncertainty set defined here constitutes a model of uncertainty over valid kernel matrices. Using a robust optimization based framework, the problem of classifier design is posed as a minimax problem. The minimax procedure is solved as a second-order-conic convex program. The formulation when applied to synthetic and protein structure data, yields significantly improved results.
Chapter 7

Efficient Algorithms for learning Robust Classifiers

Abstract

Both the formulations, based on Chance Constraint Program or Robust Optimization, for handling uncertainty in kernels are posed as minimax problems and have been solved by a novel saddle point procedure, called FSS, which has $O(1/T^2)$ convergence rate. Both of them are also solved by mirror descent algorithm (MDA) like procedure. This makes both the formulations very efficient and scalable to large scale data sets. Empirical results clearly demonstrate the validity of the claims.

7.1 Introduction

Due to the increasing interest in handling uncertain data, the need for fast and efficient training algorithms in the domains of uncertain data classification increases. In this Thesis various methods has already been discussed to build robust classifiers which can classify uncertain data more accurately. But almost all of them turns out to be solving a conic convex program more precisely Second Order Cone Program or SOCP. Though many Interior Point (IP) method based solvers are available to solve SOCP but complexities of them are around $O(n^4)$. This chapter proposes few efficient algorithms for solving kernelized classifiers proposed in Chapter 5 and Chapter 6.
7.1.1 Review of Past Work

The Chapter 5 studied the problem where the uncertainty is characterized by independent noise in each of the entries of the kernel matrix. More specifically, let $K$ is a symmetric and positive semidefinite given kernel matrix. The Chance Constraint Program (CCP) is used to solve the problem (5.3). It led to a non-convex problem (5.16) as follows:

$$
\min_{t, \alpha \in \mathbb{S}_n} \quad \frac{1}{2} t_1 - \sum_i \alpha_i \\
\text{s.t.} \quad \sum_{ij} y_i y_j \alpha_i \alpha_j K_{ij} + \kappa \sqrt{\sum_{ij} \beta_{ij} \alpha_i^2 \alpha_j^2} \leq t_1
$$

where, $\kappa = \sqrt{2 \log(1/\epsilon)}$ and $\beta$ can be calculated according to Theorem 5.2 when uncertainty has finite support. In case of Gaussian uncertainty, $\kappa = -\Phi^{-1}(\epsilon)$ and $\beta_{ij} = \sigma_{ij}^2$. If both $K, \beta$ are symmetric psd matrices then the above formulation is equivalent to formulation (7.1) (please see theorem 5.6 for details)

$$\begin{align*}
\text{RSVM}_{\text{SOCP}} \quad \min_{t_1, \theta, \varphi, \alpha \in \mathbb{S}_n} \quad & \frac{1}{2} t_1 - \sum_i \alpha_i \\
\text{s.t.} \quad & \kappa \|\beta^{\frac{1}{2}} v\| \leq t_1 - t_2 \\
& \|Y(K)^{\frac{1}{2}} \alpha\|_2 \leq t_2 \\
& \alpha_i^2 \leq \varphi_i
\end{align*}
$$

Note that this formulation is second order cone representable and hence can be solved as an Second Order Cone Program (SOCP). This will be denoted by $\text{RSVM}_{\text{SOCP}}$ ($\text{RSVM}_{\text{SOCP}}^{(g)}$ for Gaussian uncertainty)

Chapter 6 also studied the same problem where uncertainty in the kernel matrix is modeled by a convex and bounded uncertainty set which encompasses possible realizations of $K$. The realizations are modeled by an affine combination of a finite set of fixed kernels. It employed a Robust Optimization (RO) approach where instead of the probabilistic description of the uncertainty, a geometric description is used. This led to the following minimax problem (already
Chapter 7. Efficient Algorithm for learning Robust Classifiers

stated in (6.9))

\[
\max_{\alpha \in \mathcal{S}_n} \min_{\eta \in \mathcal{B}_p(\kappa)} \ - \frac{1}{2} \alpha^\top K Y \alpha - \frac{1}{2} \alpha^\top Y \sum_{l=1}^{L} (\eta_l K_l) Y \alpha + \alpha^\top e
\]

(7.2)

where

\[
\mathcal{B}_p(\kappa) = \{ \eta : ||\eta||_p \leq \kappa \}
\]

(7.3)

For the special case \( p = 2 \), the problem (7.2) can be stated as a Second Order Cone Program (SOCP) as follows:

\[
\begin{align*}
\text{(USSVM}_{\text{SOCP}}) \quad \min_{\alpha \in \mathcal{S}_{n,d}, \delta} \quad & \frac{1}{2} \delta + \frac{1}{2} \alpha^\top K Y \alpha - \alpha^\top e \\
\text{s.t.} \quad & ||\alpha||_2 \leq t \\
& \alpha^\top K_l Y \alpha \leq a_l, \forall l = 1, \ldots, L
\end{align*}
\]

(7.4)

and henceforth will be denoted by \( \text{USSVM}_{\text{SOCP}} \). (7.1) and (7.4) can be solved by Interior point (IP) algorithms (e.g. CPLEX, MOSEK, Sedumi).

### 7.1.2 Contribution

In this chapter, similar to RO approach the problem in (7.1) is written as minimax problem when \( \beta \) is positive definite matrix. As for both cases the minimization problem is a minimization of a convex function on a convex set and the maximization problem can be solved as a SVM. This kind of problems can be solved efficiently even for large scale data by an oracle type Mirror Descent Algorithm (MDA) [19, 20]. Moreover, both the approach can lead to an optimization problem which could be cast as a saddle point (minimax) problem. Due to favorable conditions satisfied by the problem, it can use a novel algorithm which uses only first order information, yet proved to decrease the initial error by a factor of \( O(1/T^2) \), after \( T \) iterations.
Chapter 7. Efficient Algorithm for learning Robust Classifiers

7.1.3 Structure of the Chapter

The MDA approach for designing robust SVMs is discussed in Section 7.2. Section 7.3 presents the saddle point algorithm and discusses its application to the minimax problem. The results of the computational study are described in Section 7.4.

7.2 Efficient Algorithm based on Mirror Descent Algorithm (MDA)

This section presents the mirror descent based algorithm for efficiently solving the problem 7.1 and 7.2. The algorithm requires only first order information, i.e. it uses only gradient projection, and has $O\left(\frac{1}{\sqrt{T}}\right)$ convergence rate. First general algorithm is presented and then the convergence of the algorithm is also presented. This section begins by rewriting the minimization problem 7.1 as a minimax problem. To this end, the following theorem is stated.

7.2.1 Mirror Descent Algorithm

The Mirror Descent Algorithm (MDA) [19, 20] solves a convex program:

$$
\min_{x \in \mathcal{X}} f(x)
$$

(7.5)

where

1. $\mathcal{X} \subset \mathbb{R}^n$ is a convex compact set.

2. The objective function $f : \mathcal{X} \to \mathbb{R}$ is a convex Lipschitz continuous function on $\mathcal{X}$, with respect to a fixed given norm $\| \cdot \|$, i.e.

$$
\exists L_f, \text{ such that } |f(x) - f(x')| \leq L_f \|x - x'\| \quad \forall x, x' \in X
$$

3. There exists an oracle which on a given $x \in \mathcal{X}$ computes $f(x)$ and $f'(x) \partial f(x)$. 
Sub-gradient Projection Algorithm (SPA)  
For such problems a classical algorithm is the Sub-gradient Projection Algorithm (SPA) \([44]\), which generates iteratively the sequence \(\{x_t\}\) via:

\[
x_{t+1} = \Pi_X (x_t - \tau(t)f'(x_t)), \quad \text{Starting from } x_1 \in X
\]

where

- \(\tau(t) > 0\) is a step-size
- \(\Pi_X (x') = \text{argmin}_{x \in X} \|x - x'\|^2\) is a standard projection on \(X\) with Euclidean geometry.
- \(f'(x)\) is a subgradient of \(f\) at \(x\)

The SPA can be rewritten equivalently as \([20]\)

\[
x_{t+1} = \text{argmin}_{x \in X} \left\{ \langle x, f'(x) \rangle + \frac{\|x - x'\|^2}{2\tau(t)} \right\}
\]

Setup for Mirror Descent Algorithm  
The main idea of Mirror Descent Algorithm (MDA) is to replace the distance function based on Euclidean norm by a general distance-like function \(\text{Dist}(x,x')\), where \(\text{Dist}(x,x')\) should maintain the property that

- \(\text{Dist}(x,x') \geq 0\)
- \(\text{Dist}(x,x') = 0 \iff x = x'\)

A possible way to construct such a distance-like function is by defining potential function \(\psi : X \rightarrow \mathbb{R}\) such that

- \(\psi\) must be continuously differentiable strongly convex on \(X\)
- The modulus of strong convexity of \(\psi\) with respect to norm \(\| \cdot \|\) is \(\rho > 0\), i.e.

\[
\langle \nabla \psi(x) - \nabla \psi(x'), x - x' \rangle \geq \rho \|x - x'\|^2, \quad \forall x, x' \in X
\]

Then \(\text{Dist}\) can be defined as Bregman Divergences generated by \(\psi\), i.e.

\[
\text{Dist}(x,x') = B_\psi(x,x') = \psi(x) - \psi(x') - \langle x - x', \nabla \psi(x') \rangle
\]
Algorithm  With this choice of $\text{Dist}$ the basic iteration step becomes

$$x^{t+1} = \arg\min_{x \in \mathcal{X}} \left\{ \langle x, f'(x') \rangle + \frac{1}{\tau(t)} \text{Dist}(x, x') \right\} \tag{7.8}$$

For suitable definition of $\text{Dist}$ one can easily get a closed form solution of $x^{t+1}$. Otherwise, this iteration scheme is also equivalent (see [20]) to the following two steps procedure. (starting with $y_1 \in \text{dom} \nabla^+ \psi$)

1. $x' = \nabla \psi^*(y')$
2. $y' = \nabla \psi(x') - \tau(t)f'(x')$

Hence sequence $\{x'\}$ is generated by $x^{t+1} = \nabla^+ \psi(\nabla \psi(x') - \tau(t)f'(x')) \tag{7.9}$

Where the conjugate function $\psi^*$ is defined as $\psi^*(y) = \max_{x \in \mathcal{X}} \{ \langle x, y \rangle - \psi(x) \}$. If this iteration goes for $t$ steps then the obtained minimum is $\min_{1 \leq t \leq T} f(x')$.  

Convergence analysis  This procedure (7.9) results into a fast convergent algorithms for solving (7.5). Here few results of convergence analysis is listed down (for details see in [20]).

**Theorem 7.1.** Let $\{x^t\}$ be the sequence generated from a starting point $x^1 \in \text{int} \mathcal{X}$ by the MDA procedure outlined in (7.9) with Dist defined in (7.7). Let at $x^* \in \mathcal{X}$ the minimum of $f$ is attained, i.e, $f(x^*) = \min_{x \in \mathcal{X}} f(x)$, then for every $t \geq 1$

**Result 7.1.**

$$\min_{1 \leq t \leq T} f(x^t) - f(x^*) \leq \frac{\text{Dist}(x^*, x^1) + 2\rho^{-1} \sum_{t=1}^{T} \tau(t)^2 \|f'(x')\|^2}{\sum_{t=1}^{T} \tau(t)} \tag{7.10}$$

**Result 7.2.** In particular, the method converges i.e. $\lim_{t \to \infty} \min_{1 \leq t \leq T} f(x^t) - f(x^*) = 0$ if the step size sequence $\{\tau(t)\}$ satisfies

- as $t \to \infty$, $\sum_{t=1}^{T} \tau(t) \to \infty$ and
- as $t \to \infty$, $\tau(t) \to 0$
Chapter 7. Efficient Algorithm for learning Robust Classifiers

**Result 7.3.** Moreover with the chosen step size

\[ \tau(t) = \sqrt{\frac{2\rho \text{Dist}(x^*, x^1)}{L_f^2 t}} \]  

(7.11)

the estimated efficiency is

\[ \min_{1 \leq i \leq t} f(x^i) - f(x^*) \leq L_f \sqrt{\frac{2\text{Dist}(x^*, x^1)}{\rho t}} \]  

(7.12)

One can approximately estimate \( \text{Dist}(x^*, x^1) \) by \( \max_{x \in X} B_\psi(x, x^1) \).

**Result 7.4.** In order to attain \( \varepsilon \) accurate solution, MDA need \( O\left(\frac{L_f^2 \text{Dist}(x^*, x^1)}{\rho \varepsilon^2}\right) \) iteration.

### 7.2.2 Solving 7.2 using MDA

The problem stated in 7.2 can be written as

\[ \min_{\eta \in \mathcal{B}_p(\kappa)} h(\eta) \]  

(7.13)

where

\[ h(\eta) = \max_{\alpha \in S_n} -\frac{1}{2} \alpha^\top Y K Y \alpha - \frac{1}{2} \alpha^\top Y \sum_{l=1}^L (\eta_l K_l) Y \alpha + \alpha^\top e \]  

(7.14)

Note that the function \( h(\eta) \) is a Lipschitz continuous function whose sub gradient is given by

\[ \partial h(\eta) = d, \text{ where } d = [d_1, \ldots, d_L]^\top \text{ and } d_l = -\frac{1}{2} \alpha^* \top Y K_l Y \alpha^* \]  

(7.15)

For a fixed \( \eta \) the inner maximization problem of (7.14) can be solved as a SVM with the kernel \( K + \sum_{i=1}^L (\eta_i K_i) \) to get \( \alpha^* \). Please note that \( \eta \) appears linearly inside \( h(\eta) \) and is constrained to lie in an ellipsoid \( \mathcal{B}_p(\kappa) = \{ \eta \mid \|\eta\|_p \leq \kappa \} \) and positive orthant \( (\eta \geq 0) \). Hence minimization problem (7.13) is minimization of a convex function on a convex set. This kind of problems can be solved efficiently even with large scale data by an oracle type Mirror Descent Algorithm (MDA) [19, 20] based procedure. As already discussed, an MDA algorithm needs a value of
Chapter 7. Efficient Algorithm for Learning Robust Classifiers

$h(\eta^t)$ and a sub-gradient of $h(\eta^t)$ as input in each iteration $t$. In this case both $h(\eta)$ and $\partial h(\eta^t)$ can easily be calculated with the help of SVM solver. The detail procedure for solving (7.2) is presented next.

Solving (7.2) when $p = 2$ with MDA leading to the following definitions.

<table>
<thead>
<tr>
<th>Potential function: $\psi(\eta) = \eta^\top \eta$ $\psi$ is strongly convex with parameter 1 in $L_2$ norm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance function: $\text{Dist}(\eta, \eta^t) = (\eta - \eta^t)^\top (\eta - \eta^t)$</td>
</tr>
<tr>
<td>Update equation (7.8): $\eta^{t+1} = \arg\min_{\eta \in \mathbb{R}_p(x), \eta \geq 0} \eta^\top \partial h(\eta^t) + \frac{1}{\tau(t)} \text{Dist}(\eta, \eta^t)$</td>
</tr>
</tbody>
</table>

(7.16)

To find a closed form solution for $\eta^{t+1}$ the following lemma can be used.

**Lemma 7.1.** Let $A$ be a symmetric positive definite matrix then the solution of

$$\arg\min_{\{x^\top Ax \leq r, x \geq 0\}} -x^\top d + \frac{(x-a)^\top A(x-a)}{s}$$

has the following closed form

$$x^* = \max \left( \frac{a + \frac{sA^{-1}d}{2}}{\sqrt{\frac{s}{2}d^\top A^{-1}d + sa^\top Aa}}}, 0 \right)$$

(7.18)

**Proof.** Using Lagrange multiplier $\sigma$ for the constraint $x^\top Ax \leq r$ the dual of the minimization problem in (7.17) is

$$\max_{\sigma \geq 0} \min_{x \geq 0} -x^\top d + \frac{(x-a)^\top A(x-a)}{s} + \sigma(x^\top Ax - r)$$
By equating derivatives to zero one gets

\[-d + \frac{2Ax}{s} - \frac{2Aa}{s} + 2\sigma Ax = 0\]

or

\[x^* = \frac{a + \frac{sA^{-1}d}{2}}{1 + s\sigma}\]  \hspace{1cm} (7.19)

From the fact that at optimality \(x^TAx = r\), one can calculate

\[(1 + s\sigma)^2 = \frac{(a + \frac{sA^{-1}d}{2})A(a + \frac{sA^{-1}d}{2})}{r}.

This completes the proof.

Using the above theorem one can make the following claim.

**Claim 7.1.** The problem, described in (7.16), has a closed form as

\[\eta^{t+1} = \max\left(\frac{\kappa(\eta^t + \tau(t)(-d))}{\sqrt{\frac{\tau(t)^2(-d)^T(-d)}{4} + \kappa^2 + \tau(t)\eta^T(-d)}}, 0\right)\]  \hspace{1cm} (7.20)

when \(d\) is defined in (7.15).

**Proof.** Replacing \(A = \text{Identity matrix, } r = \kappa^2, a = \eta^k, d = (-d)\) and \(s = \tau(t)\) in Lemma 7.1 and using the fact that \(\eta^T\eta^t = \kappa^2\) the result is achieved. \(\square\)

These observation motivates the following algorithm.

**Algorithm 1 USSVM\text{MDA} :** MDA based fast algorithm for USSVM

Initialize \(\eta^1\)

repeat

Get \(\alpha^{t'}\) by solving SVM with kernel \(\mathbf{K} + \sum_{l=1}^{L} (\eta_l \mathbf{K}_l)\)

\[\eta^{t+1} = \max\left(\frac{\kappa(\eta^t + \tau(t)(-d))}{\sqrt{\frac{\tau(t)^2(-d)^T(-d)}{4} + \kappa^2 + \tau(t)\eta^T(-d)}}, 0\right)\]

until Convergence in \(\eta\)

**Step Size**  By Theorem 7.1 the choice of step-size is guided by the term \(\text{Dist}(\eta^*, \eta^1)\), where \(\eta \in \mathcal{B}_2\) and \(\eta \geq 0\). Considering \(\eta^1 = 0, \max_{\eta \in \mathcal{B}_2} \text{Dist}(\eta, 0) = \kappa^2\). Then the candidate step-size
Chapter 7. Efficient Algorithm for learning Robust Classifiers

(see Theorem 7.1 and (7.11)) becomes

$$\tau(t) = \frac{\kappa \sqrt{2}}{L_h \sqrt{t}}$$

(7.21)

where $L_h$ is the Lipschitz constant of $h$. However this step-size estimate is not useful as $L_h$ will not be known a priori. A more pragmatic choice could be replacing $L_h$ by $\frac{\|h'(\eta)\|_\infty}{A}$ (see [19]). Using the fact that $\|h'(\xi)\|_{\infty} \leq C^2 L \max_i (\lambda_{\max}(K_i))$ the step size becomes

$$\tau(t) = \frac{\kappa \sqrt{2} A}{C^2 L \max_i (\lambda_{\max}(K_i)) \sqrt{t}}$$

(7.22)

where A is a constant. It can be shown [19] that even for this step-size an efficiency estimate of convergence is similar to the one given in Theorem 7.1.

**Convergence of USSVM_{MDA}** Using information from Theorem 7.1, Algorithm 1 will get optimum solution with $\varepsilon$ accurate optimum $\eta$ in $O^{C^4 L^2 \max_i (\lambda_{\max}^2(K_i)) \varepsilon^2}$ or $O\left(\frac{\max_i (\lambda_{\max}^2(K_i))}{\varepsilon^2}\right)$. Note that algorithm described in USSVM_{MDA} contains a SVM solver in each iteration as a gradient computing oracle. Therefore the computational complexity of USSVM_{MDA} remains

$$O\left(\frac{\max_i (\lambda_{\max}^2(K_i))}{\varepsilon^2}\right) O(SVM)$$


7.2.3 **Solving 7.1 using MDA**

The MDA procedure also applies to formulation (7.1). If $\beta$ is psd then the formulation (7.1) can be posed as a SOCP (7.1). One could recast this problem as the following mini-max problem.

**Theorem 7.2.** The formulation (7.1) is equivalent of solving

$$\min_{\xi \geq 0} \quad h(\xi)$$

$$\sqrt{\xi} \beta^{-1} \xi \leq \frac{\xi}{2}$$

(7.23)
where \( h(\zeta) = \max_{\alpha \in S_n} -\frac{1}{2} \alpha^T Y K Y \alpha + \sum_{i=1}^n \alpha_i - \sum_i \zeta_i \alpha_i^2 \) \hspace{1cm} (7.24)

where \( \zeta_i \)'s are the Lagrange multipliers of \( \alpha_i^2 \leq v_i \) in (7.1)

Proof. At the optimum of (7.1) the constraint involving \( t_1 \) and \( t_2 \) is active. Eliminating both \( t_1 \) and \( t_2 \), the resultant problem is equivalent to solving,

\[
\max_{\zeta \geq 0} \min_{\alpha \in S_n} L(\zeta, \alpha) = \frac{1}{2} \left( \alpha^T Y K Y \alpha + \kappa \sqrt{\nu^T \beta} \nu \right) - \sum_{i=1}^n \alpha_i + \sum_{i=1}^n \zeta_i (\alpha_i^2 - v_i) \hspace{1cm} (7.25)
\]

where \( \zeta_i \) is the Lagrange multiplier of the constraint \( \alpha_i^2 \leq v_i \). Using first order conditions of optimality, \( \frac{\partial L}{\partial v_i} = 0 \) one obtains

\[
\frac{\kappa}{2} (\nu^T \beta \nu)^{-\frac{1}{2}} \beta \nu + \zeta = 0 \\
\sqrt{\zeta^T \beta^{-1} \zeta} = \frac{\kappa}{2} \hspace{1cm} (7.26)
\]

Eliminating \( \nu \) and noting that at optimality the constraint of (7.23) involving \( \zeta \) is active, the proof is complete.

Note that, the function \( h(\zeta) \) is a Lipschitz continuous function whose sub gradient is given by \( \partial h(\zeta) = d = -\alpha^* \ast \alpha^* \), where for a fixed \( \zeta \) the inner maximization problem of (7.24) can be solved as a SVM with the kernel \( K + \text{diag}(\zeta) \) to get \( \alpha^* \). Please note that \( \zeta \) appears linearly inside \( h(\zeta) \) and is constrained to lie in an ellipsoid \( \delta(\beta^{-1}, \kappa) = \{ \zeta : \sqrt{\zeta^T \beta^{-1} \zeta} \leq \frac{\kappa}{2} \} \) and positive orthant \( (\zeta \geq 0) \). Hence minimization problem (7.23) is minimization of a convex
Chapter 7. Efficient Algorithm for learning Robust Classifiers

function on a convex set. Hence (7.23) can be solved with MDA using following definitions

Potential function: \[ \psi(\zeta) = \zeta^T \beta^{-1} \zeta \]

\( \psi \) is strongly convex with parameter \( \frac{1}{\lambda_{\text{max}}(\beta)} \) in L2 norm

Distance function: \[ \text{Dist}(\zeta, \zeta') = (\zeta - \zeta')^T \beta^{-1} (\zeta - \zeta') \]

Update equation (7.8): \[ \zeta^{k+1} = \arg\min_{\zeta \in \mathbb{R}^{n \times d} \backslash \{0\}, \zeta \geq 0} \left( \zeta^T \partial h(\zeta') + \frac{1}{\tau(t)} \text{Dist}(\zeta, \zeta') \right) \]

Algorithm: Algorithm 2

Step size: \( \tau(t) = \frac{\kappa A}{C^2 \sqrt{\lambda_{\text{max}}(\beta)} t} \)

Algorithm 2 RSVM_{MDA}: MDA based fast algorithm for RSVM

1. Initialize \( \zeta^1 \)
2. repeat
   - Get \( \alpha^{t'} \) by solving SVM with kernel \( K + \text{diag}(\zeta') \)
   - \[ \zeta^{t+1} = \max \left( \frac{\zeta^t + \frac{t^{(t)} \beta d}{\kappa^2} + \frac{4}{\kappa^2} \zeta^{t'} d}{\sqrt{\frac{t^{(t)} \beta d}{\kappa^2} + 1 + \frac{4}{\kappa^2} \zeta^{t'} d}}, 0 \right) \]
3. until Converge in \( \zeta \)

Convergence of RSVM_{MDA} Using information from Theorem 7.1, Algorithm 2 will get optimum solution with \( \varepsilon \) accurate optimum \( \zeta \) in \( \frac{C^4 \kappa^2 \lambda_{\text{max}}(\beta)}{2 A \varepsilon^2} \) or \( O \left( \frac{\lambda_{\text{max}}}{\varepsilon^2} \right) \). Note that algorithm described in RSVM_{MDA} contains a SVM solver in each iteration as a gradient computing oracle. Therefore the computational complexity of RSVM_{MDA} remains \( O \left( \frac{\lambda_{\text{max}}}{\varepsilon^2} \right) O(SVM) \).

7.3 An Algorithm for a special class of Convex-Concave Saddle Point Problem

This section describes a novel algorithm for a class of convex-concave saddle point problems and show that it applies to both (7.2) and (7.1). The algorithm also requires only first order
information, i.e. it uses only gradient projection, and has $O(1/T^2)$ convergence rate. The algorithm proceeds in stages, where in each stage several updates happen. At first, the general algorithm is presented and then focus goes to the special case, called FSS, where the step size is held constant over each stage.

### 7.3.1 Assumptions on the Saddle Point Problem

Let $\mathcal{U}$ be a closed convex set in Euclidean space $\mathcal{E}$, $\| \cdot \|$ be a norm on $\mathcal{E}$, and $\omega(u) : \mathcal{U} \rightarrow \mathbb{R}$ be a function. One can say that $\omega(\cdot)$ is a distance-generating function (d.g.f.) for $\mathcal{U}$ compatible with $\| \cdot \|$, if the following properties hold.

- $\omega$ is convex and continuous on $\mathcal{U}$
- there exist $\omega'(u) \in \partial \omega(u)$, which is continuous on the set $\mathcal{U}^0 = \{ u : \partial \omega(u) \neq \emptyset \}$ and is strongly convex, modulus 1 w.r.t. $\| \cdot \|$ such that
  \[
  \langle \omega'(u) - \omega'(u'), u - u' \rangle \geq \| u - u' \|^2 \quad \forall u, u' \in \mathcal{U}^0. 
  \]  
  (7.27)

Assume that

- $\mathcal{X}$ is a closed and bounded convex subset in a Euclidean space $\mathcal{E}_x$, and $\mathcal{V}$ is a closed convex subset in a Euclidean space $\mathcal{E}_v$
- $\mathcal{E}_x$, $\mathcal{E}_v$ are equipped with norms $\| \cdot \|_x$, $\| \cdot \|_v$, the conjugate norms being $\| \cdot \|_{x^*}$, $\| \cdot \|_{v^*}$
- $\mathcal{X}$ is equipped with a d.g.f. $\omega_x(\cdot)$ compatible with $\| \cdot \|_x$, and $\mathcal{E}_v$ is equipped with a d.g.f. $\omega_v(\cdot)$. $x_\omega$ denotes the minimizer of $\omega_x(\cdot)$ on $\mathcal{X}$ (note that $x_\omega \in \mathcal{X}^0$) and set
  \[
  \Omega_x = \max_{x \in \mathcal{X}} \omega_x(x) - \min_{x \in \mathcal{X}} \omega_x(x). 
  \]  
  (7.28)

Assuming that the minimizer of $\omega_v(\cdot)$ is the origin in $\mathcal{E}_v$, one can set

- $\Omega_v = \max_{\| v \| \leq 1} \omega_v(v) - \min_{v \in \mathcal{E}_v} \omega_v(v)$.  
  (7.29)
Chapter 7. Efficient Algorithm for learning Robust Classifiers

Hence the following saddle point problem need to be solved

\[
\text{SadVal} = \min_{x \in \mathcal{X}} \max_{v \in \mathcal{V}} \phi(x, v), \tag{7.30}
\]

where \( \phi(\cdot, \cdot) \) satisfies the following assumptions:

A.1. \( \phi(x, v) : \mathcal{X} = \mathcal{X} \times \mathcal{V} \rightarrow \mathbb{R} \) is continuously differentiable function which is convex in \( x \in \mathcal{X} \) and is strongly concave, modulus \( \theta > 0 \) w.r.t. \( \| \cdot \|_v \), in \( v \in \mathcal{V} \), that is,

\[
\langle \phi'_v(x, v') - \phi'_v(x, v), v - v' \rangle \geq \theta \| v - v' \|_v^2 \forall (x \in \mathcal{X}, v,v' \in \mathcal{V}) \tag{7.31}
\]

A.2. \( \nabla \phi(\cdot, \cdot) \) is Lipschitz continuous on \( \mathcal{X} = \mathcal{X} \times \mathcal{V} \)

A.3. \( \phi(x, v) \) is affine in \( x \): \( \phi(x, v) = x^T a(v) + g(v) \).

In the sequel, please set

\[
\overline{\phi}(x) = \max_{v \in \mathcal{V}} \phi(x, v), \quad \underline{\phi}(v) = \min_{x \in \mathcal{X}} \phi(x, v),
\]

so that \( \overline{\phi} \) is a continuous convex function on \( \mathcal{X} \), \( \underline{\phi} \) is a continuous strongly concave, modulus \( \theta \) w.r.t. \( \| \cdot \|_v \), function on \( \mathcal{V} \), and

\[
\min_{x \in \mathcal{X}} \overline{\phi}(x) = \text{SadVal} = \max_{v \in \mathcal{V}} \underline{\phi}(v) \tag{7.32}
\]

and the set of saddle points of \( \phi \) on \( \mathcal{X} \times \mathcal{V} \) is \( \mathcal{X}_s \times \{ v_s \} \), where \( \mathcal{X}_s = \text{argmin}_{\mathcal{X}} \overline{\phi} \), and \( v_s = \text{argmin}_{\mathcal{V}} \underline{\phi} \). Finally, \( \varepsilon_{\text{sad}}(z), z \in \mathcal{X} \) denotes the natural saddle point proximity measure:

\[
\varepsilon_{\text{sad}}(x, v) = \overline{\phi}(x) - \underline{\phi}(v) = \left[ \overline{\phi}(x) - \min_{\mathcal{X}} \overline{\phi} \right] + \left[ \max_{\mathcal{V}} \underline{\phi} - \underline{\phi}(v) \right].
\]

7.3.2 Algorithm for solving Convex-Concave saddle point problem

The algorithm presented here is a particular case of algorithm presented in [45].
Chapter 7. Efficient Algorithm for learning Robust Classifiers

Preliminaries

Let

\[ G(x, v) = \begin{bmatrix} G_x(x, v) := \frac{\partial \phi(x, v)}{\partial x} = a(v); G_v(x, v) := -\frac{\partial \phi(x, v)}{\partial v} \end{bmatrix} \]

be the monotone operator associated with the saddle point problem (7.30). By A.1, this operator is Lipschitz continuous, and, its \(x\)-component depends solely on \(v\). As a result, one can specify the partial Lipschitz constants \(L_{xv}, L_{vv}\) such that

\[
\forall (x, x' \in \mathcal{X}, v, v' \in \mathcal{V}) : \\
\|G_x(x, v) - G_x(x', v')\|_{x,*} \leq L_{xv}\|v - v'\|_v; \|G_v(x, v) - G_v(x', v')\|_{v,*} \leq L_{xv}\|x - x'\|_x; \\
\|G_v(x, v) - G_v(x', v')\|_{v,*} \leq L_{vv}\|v - v'\|_v.
\]

Execution of Algorithm MPb is split into stages. At the beginning of stage \(s = 0, 1, \ldots\) algorithm has a disposal positive \(R_s\) and a point \(\tilde{v}_s \in \mathcal{V}\) such that

\[ \|\tilde{v}_s - v_*\|_v \leq R_s / 2. \quad (I_s) \]

These data define the quantities

\[
\mathcal{X}_s = \{(x; v) \in \mathcal{X} : \|v - \tilde{v}_s\|_v \leq R_s\}, \quad L_s = 2L_{xv}\sqrt{\Omega_x\Omega_v}R_s + L_{vv}\Omega_vR_s^2, \quad \tau(s) = 1 / L_s \\
\alpha_s = \frac{[L_{xv}\sqrt{\Omega_x\Omega_v}R_s]}{L_s}, \quad \beta_s = \frac{[L_{xv}\sqrt{\Omega_x\Omega_v}R_s + L_{vv}\Omega_vR_s^2]}{L_s} = 1 - \alpha_s, \\
\omega_s(x, v) = \left[ \alpha_s + \frac{\beta_s}{\Omega_v}[v - \tilde{v}_s] / R_s \right], \\
\|\langle x, v \rangle\|_{(s)} = \sqrt{\frac{\alpha_s}{\Omega_v} \|x\|_x^2 + \frac{\beta_s}{\Omega_v R_s^2} \|v\|_v^2} \Rightarrow \\
\|\langle \xi, \eta \rangle\|_{(s),*} = \sqrt{\frac{\alpha_s}{\Omega_v} \|\xi\|_{\xi,*}^2 + \frac{\beta_s R_s^2}{\Omega_v} \|\eta\|_{\eta,*}^2}.
\]
Lemma 7.2. $\omega_s$ is a d.g.f. for $\mathcal{X}$ compatible with the norm $\| \cdot \|_{(s)}$, and

\[
\begin{align*}
(a) \quad & \arg\min_{\mathcal{X}} \omega_h(\cdot) = (x_0, \bar{v}_s) \in \mathcal{X}_s \text{ extremalmax}_{\mathcal{X}_s} \omega_h(\cdot) - \min_{\mathcal{X}_s} \omega_h(\cdot) \leq 1, \\
(b) \quad & \forall (z, z' \in \mathcal{X}) : \| G(z) - G(z') \|_{(s)}, \leq L_s \| z - z' \|_{(s)}.
\end{align*}
\]

(7.36)

Proof of this lemma (and of all subsequent statements of this section) is present in Appendix D.

In the sequel, set

\[
V_z^s(u) = \omega_h(u) - \omega_h(z) - (\omega'_h(z), u - z) \quad [z \in \mathcal{X}^0, u \in \mathcal{X}]
\]

(7.37)

Stage $s$ of Algorithm MPb At stage $s$ carry out

\[
N_s = \text{Ceil} \left( \frac{64L_{xy} \sqrt{\Omega_s \Omega_{s-1} R_s^{-1}} + 32L_{xy} \Omega_s}{\theta} \right)
\]

(7.38)

steps of the following recurrence:

1. Initialization: Set $z_{1,s} = (x_0, \bar{v}_s) = \arg\min_{\mathcal{X}} \omega_h(\cdot)$.

2. Step $t = 1, 2, \ldots, N_s$: Given $z_{t,s} \in \mathcal{X}^0$, choose $\tau(t, s) \geq \tau(s)$, compute

\[
\begin{align*}
wt_{t,s} &= \arg\min_{\mathcal{X}} \left\{ (\tau(t, s) G(z_{t,s}) - \omega'_h(z_{t,s}), u) + \omega_h(u) \right\} \\
z_{t+1,s} &= \arg\min_{\mathcal{X}} \left\{ (\tau(t, s) G(w_{t,s}) - \omega'_h(z_{t,s}), u) + \omega_h(u) \right\} \\
\delta_{t,s} &= (\tau(t, s) G(w_{t,s}), w_{t,s} - z_{t+1,s}) - V_{z_{t+1,s}}^s
\end{align*}
\]

(7.39)

and pass to step $t + 1$, provided $t < N_s$. When $t = N_s$, the approximate solution to (7.30) built in course of $s$ stages is defined as

\[
(x^s, v^s) = \left[ \sum_{t=1}^{N_s} \tau(t, s) \right]^{-1} \sum_{t=1}^{N_s} \tau(t, s) wt_{t,s},
\]

(7.40)

set

\[
\bar{v}^s+1 = v^s, R_{s+1} = R_s / 2
\]
and pass to stage $s + 1$.

The convergence properties of the algorithm are given by the following

**Proposition 7.1.** Let assumptions A.1-3 take place, let $R_0 > 0$ and $\bar{v}_0 \in \mathcal{V}$ satisfying (I0) be given, and let the choice of step sizes $\tau(t, s) \geq \tau(s)$ ensure that $\delta_{t,s} \leq 0$ (to this end, it suffices to set $\tau(t, s) = \tau(s)$). Then, for every $s$, (I$s$) takes place, and

$$
\varepsilon_{sad}(x^s, v^s) \leq \theta R_0^2 2^{-2s-5},
$$

while the total number $M_s = \sum_{i=0}^{s} N_i$ of steps of the algorithm needed to build $(x^s, v^s)$ admits the bound

$$
M_s \leq O(1) \left[ \frac{L_{xy} \sqrt{\Omega_x \Omega_v}}{\theta R_0} 2^s + \frac{L_{xy} \Omega_v + \theta}{\theta} (s+1) \right].
$$

In particular, setting

$$
s^* = \max \left\{ s : \frac{L_{xy} \sqrt{\Omega_x \Omega_v}}{L_{xy} \Omega_v + \theta} 2^s \leq (s+1)R_0 \right\},
$$

$$
s \leq s^* \Rightarrow \varepsilon_{sad}(x^s, v^s) \leq \theta R_0^2 2^{-2(s+2)} \quad \text{and} \quad M_s \leq O(1) \frac{L_{xy} \Omega_v \theta}{\theta R_0} (s+1)
$$

$$
s > s^* \Rightarrow \varepsilon_{sad}(x^s, v^s) \leq O(1) \frac{L_{xy} \Omega_v \theta}{\theta M_s^2} \quad \text{and} \quad M_s \leq O(1) \frac{L_{xy} \sqrt{\Omega_x \Omega_v}}{\theta R_0} 2^s.
$$

Proof. See Appendix D. \qed

The above proposition points us to the convergence rate of $O(1/M_s^2)$ when number of stages is sufficiently high.

In this Thesis, hereafter FSS considers the fixed step size per stage, i.e. $\tau(t, s) = \tau(s)$, version of the algorithm. For such a choice computation of $\delta_{t,s}$ in (7.39) is not required. In the following section their applicability to the problem at hand is described.

### 7.3.3 Application of FSS Algorithm to (7.2)

This section discusses the application of FSS algorithm for solving formulation (7.2). It is strictly concave in $\alpha$, provided $\bar{K}$ is positive definite, and affine in $\eta$. The domain of $\alpha$ and $\eta$ are non-empty convex sets. Clearly the formulation obeys all the assumptions, A.1-3, of the
FSS procedure. In order to be consistent with the notation of FSS procedure let’s define the following map where LHS denotes quantities involving formulation (7.2) and the right hand side corresponds to the saddle point procedure detailed in the previous section. In particular following has been used \( \eta \to x, \alpha \to v, YKQ \to \overline{Q}, YK_iY \to Q_i, E_l = \mathbb{R}^L \), leading to the following definitions.

\[
\begin{align*}
\mathcal{V} &= \{v \in \mathbb{R}^n : C \geq v_i \geq 0, \sum v_i = 0\} \subset \mathbb{R}^n \\
\mathcal{X}^{-} &= \{x \in \mathbb{R}^L : x \geq 0, ||x||_2 \leq 1\} \subset \mathbb{R}^L \\
\phi(x, v) &= v^T \overline{Q}v - \sum_{l=1}^{L} x_l (\frac{1}{2} v^T Q_l v) + v^T e \\
G_x(x, v) &= -d = [d_1, \ldots, d_L]^T, d_l = \frac{1}{2} v^T Q_l v \\
G_v(x, v) &= (\sum_{l=1}^{L} x_l Q_l)v - \epsilon_1, \epsilon_1 = [1, \ldots, 1]^T \\
||x||_2 &= ||x||_2, w_x(x) = \frac{1}{2} ||x||^2, \Omega_x = \frac{1}{2} \\
||v||_2 &= ||v||_2, w_v(v) = \frac{1}{2} ||v||^2, \Omega_v = \frac{1}{2} \\
L_{sv} &= 2C\kappa \sqrt{L_{\max}(\lambda_{\max}(Q_l))} \\
L_{xy} &= \kappa \sqrt{L_{\max}(\lambda_{\max}(Q_l))} + \lambda_{\max}(\overline{Q}) R_0 = 2C \sqrt{n}
\end{align*}
\]

**Proposition 7.2.** Computing the right-hand sides in every step in (7.39) is equivalent to solve problems of the form

\[
\begin{align*}
(a) \quad & x_+ = \arg\min_{x \in \mathcal{X}} \left[ \frac{1}{2} (x - \bar{x})^T (x - \bar{x}) - p^T x \right], \\
(b) \quad & v_+ = \arg\min_{v \in \mathcal{V}} \left[ \frac{1}{2} (v - \bar{v})^T (v - \bar{v}) - q^T v \right], \quad (7.44)
\end{align*}
\]

where \( \bar{x} \in \mathcal{X}, \bar{v} \in \mathcal{V} \) are solution in previous iteration and \( p = -\frac{\Omega_x \tau(t,s)}{\alpha_s} G_x(\bar{x}, v) \) and \( q = -\frac{\Omega_v \tau(t,s)}{\beta_s} R_s^T G_v(\bar{x}, \bar{v}) \). The point \((\bar{x}, \bar{v})\) are intermediate points in (7.39).

**Proof.** See that one have \( \omega (x, v) = \frac{\mu_s}{2} x^T x + \frac{\nu_s}{2} v^T v \) with certain positive \( \mu_s, \nu_s \) (see (7.35)).
hence a computation of the type of (7.39), that is

$$\{ \bar{z} = (\bar{x}, \bar{v}) \in Z^o, \tau > 0, G(\bar{x}, \bar{v}) \}$$

$$\Rightarrow z_+ = (x_+, v_+) := \text{argmin}_{u \in Z} \{ \langle \tau G(\bar{x}, \bar{v}) - \omega_x(z), u \rangle + \omega_y(u) \}$$  \hspace{1cm} (!!)

reduces to

$$x_+ = \text{argmin}_{x \in X} \{ x^T [\tau G_x(\bar{x}, \bar{v}) - \mu_x \beta^{-1} \bar{x}] + \frac{\mu_x}{2} x^T \beta^{-1} x \}$$

$$= \text{argmin}_{x \in X} \left\{ -p^T x + \frac{1}{2} (x - \bar{x})^T (x - \bar{x}) \right\},$$

and

$$v_+ = \text{argmin}_{v \in Y} \{ v^T [\tau G_v(\bar{x}, \bar{v}) - v_s \bar{v}] + \frac{v_s}{2} v^T v \}$$

$$= \text{argmin}_{v \in Y} \left\{ -v^T q + \frac{1}{2} (v - \bar{v})^T (v - \bar{v}) \right\},$$

$$\square$$

**Update equation for x and v**  
Assuming that algorithm start with a feasible point, i.e. \( x^0 \geq 0 \) and \( \|x^0\|_2 = \kappa \), then \( x^{i+1} = \kappa \frac{p + x^i}{\sqrt{p^T p + 2 p^T x^i + \kappa^2}} \). The computation steps involving \( y \) is solved by projecting a vector onto the constraint set of the Dual SVM problem. The problem can be solved by a Quadratic program. However a line search procedure described in Appendix E is proposed in this Thesis. That results in considerable saving of computation time.

The solution of formulation (7.2) by FSS procedure will be referred to as \( \text{USSVM}_{\text{FSS}} \).

### 7.3.4 Application of FSS Algorithm to (7.1)

The FSS procedure is fairly general and also applies to formulation (7.1). According to Theorem 7.2 if \( \beta \) is psd then the formulation (7.1) can be posed as the following mini-max problem.

$$\min_{\zeta \geq 0} \max_{\alpha \in S_n} \begin{cases} -h(\zeta, \alpha) \\ \sqrt{\zeta} \beta^{-1} \zeta \leq \frac{\kappa}{2} \end{cases}$$ \hspace{1cm} (7.45)

where \( h(\zeta, \alpha) = \frac{1}{2} \alpha^T Y^T Y \alpha - \sum_{i=1}^{n} \alpha_i + \sum_{i} \zeta_i \alpha_i^2 \) \hspace{1cm} (7.46)
Note that, the function $h(\zeta, \alpha)$ is Lipschitz continuous. It is linear in $\zeta$ and strongly convex in $\alpha$ as long as $\mathbf{K}$ is positive definite. Both $\zeta$ and $\alpha$ lie in a convex and compact set. In principle the FSS procedure applies and could be an interesting alternative to the SOCP procedure discussed in Chapter 5.

As before the connection between the variables of FSS procedure and (7.45) is described explicitly.

$\zeta \rightarrow x, \alpha \rightarrow v, YK \mathbf{y} \rightarrow Q$ using this mapping the following definition holds.

$$\mathcal{X} = \{ x \in \mathbb{R}^n : x \geq 0, x^T \beta^{-1} x \leq 1 \} \subset \mathbb{E}_l = \mathbb{R}^l$$

$$\mathcal{Y} = \{ v \in \mathbb{R}^n : C \geq v_i \geq 0, \sum_i v_i y_i = 0 \} \subset \mathbb{E}_n = \mathbb{R}^n$$

$$\phi(x, v) = -\frac{1}{2} v^T Q v + \sum_{i=1}^n v_i - \sum_i x_i v_i^2$$

$$G_x(x, v) = -[v_1^2; \ldots; v_n^2]$$

$$G_v(x, v) = 2[x_1 v_1; \ldots; x_n v_n] + Q v - [1; \ldots; 1]$$

$$\|x\|_x = \sqrt{x^T \beta^{-1} x} \text{ and } w_x(x) = \frac{1}{2} x^T \beta^{-1} x \text{ and } x_w = 0 \text{ and } \Omega_x = \frac{1}{2} \frac{(\mathbf{K})^2}{2}$$

$$\|v\|_v = \sqrt{v^T v} \text{ and } w_v(y) = \frac{1}{2} v^T v \text{ and } \Omega_v = \frac{1}{2} \text{ and } \theta = \lambda_{\min}(Q)$$

$$L_{xv} = 2C\lambda_{\max}^{\frac{1}{2}}(\beta) \text{ and } L_{vv} = 2\lambda_{\max}(Q)$$

$$R_0 = 2C \sqrt{n}$$

(7.47)

**Proposition 7.3.** Computing the right-hand sides in every step in (7.39) is equivalent of solving problems of the form

(a) $x_+ = \arg\min_{x \in \mathcal{X}} \left[ \frac{1}{2} (x - \bar{x})^T \beta^{-1} (x - \bar{x}) - p^T x \right]$,

(b) $v_+ = \arg\min_{v \in \mathcal{Y}} \left[ \frac{1}{2} (v - \bar{v})^T (v - \bar{v}) - q^T v \right]$,

(7.48)

where $\bar{x} \in \mathcal{X}^*$, $\bar{v} \in \mathcal{Y}^*$ are solution in previous iteration and $p = -\frac{\Omega_x(\tau,x)}{\alpha_x} G_x(\bar{x}, \bar{v})$ and $q = -\frac{\Omega_v(\tau,v)}{\beta_v} R_0^2 G_v(\bar{x}, \bar{v})$. 

Chapter 7. Efficient Algorithm for learning Robust Classifiers

Proof. Similar to the proof of proposition 7.2.

Update equation for $x$ and $v$ Starting with a feasible point, i.e. $x^0 \geq 0$ and $(x^0)^T \beta^{-1} x^0 = (\frac{\kappa}{2})$ (for example $x^0 = (\frac{\kappa}{2}) \sqrt{\lambda_{max}(\beta)} e_{max}$), then $x^{k+1} = (\frac{\kappa}{2}) \frac{\beta b^k}{[(b^k)^T \beta b^k]^{1/2}}$, where $b^k = p^k + \beta^{-1} x^k$. The computation steps involving $v$ is solved by projecting a vector onto the constraint set of the Dual SVM problem. The problem can be solved by a Quadratic program. However this Thesis proposes a line search procedure described in Appendix E which results in considerable saving of computation time. The solution of formulation (7.1) by FSS procedure will be referred to as $\text{RSVM}_{\text{FSS}}$.

7.3.5 Discussion

A minimax problem

$$\min_{x \in X} \max_{y \in Y} \phi(x, y)$$

can be solved as follows

$$\min_{x \in X} g(x) = \max_{y \in Y} \phi(x, y)$$ (P)

Problem (P) is in general non-smooth. However under certain favourable conditions, it can be solved with $O(1/T^2)$ convergence by an algorithm due to [46]. One of the conditions required for applying the nesterov procedure is $\nabla_x g(x)$ should be Lipchitz continuous. For the problem (7.2), this condition is not necessarily satisfied. The saddle point procedure just discussed does not require this assumption and hence directly applies to the problem at hand.

7.4 Experimental Evaluation

This section presents experimental evaluation of the formulations discussed before. In particular it would be interesting to explore the following questions.

1. Comparison of $\text{USSVM}_{\text{MDA}}$ and $\text{USSVM}_{\text{FSS}}$ against $\text{USSVM}_{\text{SOCP}}$ and also of $\text{RSVM}_{\text{MDA}}$ and $\text{RSVM}_{\text{FSS}}$ against $\text{RSVM}_{\text{SOCP}}$

2. Scalability of MDA procedure.
3. Convergence and scalability of the FSS procedure

The section is organized as follows. The section begins by a discussion of first issue in Section 7.4.2. The experimental verification of the convergence rate of FSS procedure is discussed in Section 7.4.3. Next the scalability of MDA and FSS procedure is discussed in Section 7.4.4.

7.4.1 Datasets

Following synthetic datasets is created to test the generalization properties of the proposed formulations.

- **Synthetic Data** The datasets $\mathbf{DK}_G(S,N,L), \mathbf{DK}_U(S,N,L), \mathbf{DK}_B(S,N,L)$ have been created with help of the generating procedure described in Section 6.4.1.

**Platform and package** The algorithms USSVM: $USSVM_{SOCP}, USSVM_{MDA}$ and $USSVM_{FSS}$ and RSVM: $RSVM_{SOCP}, RSVM_{MDA}$ and $RSVM_{FSS}$ have been implemented in Matlab with the help of Sedumi\(^1\) [33]. libSVM\(^2\) has been used as an SVM solver. All the experiments have been performed on a 64 bits Linux PC with 8 Intel Xeon 2.66 GHz processors and 16GB of RAM.

7.4.2 Comparison of Generalization Error

This section have performed a thorough experimental analysis of the proposed formulations measuring their generalization performances. The proposed formulations along with their respective algorithms have been compared here using the four metrics described in section 5.4.2.

**Datasets and Kernels** Datasets $\mathbf{DK}_G(S,N,L), \mathbf{DK}_U(S,N,L), \mathbf{DK}_B(S,N,L)$ were created with $S = 10, N = 250, L = 200$, as described in Section 6.4.1. The uncertain kernels were generated by using $R = 100$ and $\kappa = 1$(see Section 6.3). Note that $\mathbf{DK}_G$ refers to the Gaussian case, $\mathbf{DK}_U$ refers to the Uniform case and $\mathbf{DK}_B$ refers to the Beta case.

---

1http://sedumi.ie.lehigh.edu/
2http://www.csie.ntu.edu.tw/ cjlin/libsvm/
For all the metrics, 5-fold cross-validation was performed on all the 10 datasets corresponding to each distribution. The hyper-parameters \((C \text{ and } \epsilon)\) for each classifier, were chosen using a grid search mechanism from the set \(C = \{0.1, 1, 5, 10, 50, 100, 200, 500\}\) and \(\epsilon = \{0.05 + 0.05 \times \text{step} | \text{step} = 0, \ldots, 9\}\). For each metric, the cross-validation accuracy, 100\((1 - \text{ErrorMeasure})\)\%, averaged over 10 datasets for various distributions, are reported in Table 7.1.

Table 7.1: Cross-validation accuracy (%) obtained with \(USSVM_{SOCP}, USSVM_{MDA}, USSVM_{FSS}, RSVM_{SOCP}, RSVM_{MDA}, RSVM_{FSS}\) and Nominal-SVM using RobustErr(RE, (5.21)), MajErr(ME, (5.20)) and NomErr(NE, (5.22)). All values reported here are 100\((1 - \text{Errormeasure})\)\%.

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>USSVM_{SOCP}</th>
<th>USSVM_{MDA}</th>
<th>USSVM_{FSS}</th>
<th>RSVM_{SOCP}</th>
<th>RSVM_{MDA}</th>
<th>RSVM_{FSS}</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DK_G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>89.00</td>
<td>89.00</td>
<td>88.60</td>
<td>89.00</td>
<td>88.90</td>
<td>89.00</td>
<td>78.70</td>
</tr>
<tr>
<td>RE</td>
<td>75.60</td>
<td>75.60</td>
<td>76.20</td>
<td>66.90</td>
<td>64.10</td>
<td>65.50</td>
<td>26.10</td>
</tr>
<tr>
<td>NE</td>
<td>89.40</td>
<td>89.40</td>
<td>89.60</td>
<td>89.40</td>
<td>89.20</td>
<td>89.40</td>
<td>90.90</td>
</tr>
<tr>
<td>SE</td>
<td>88.94</td>
<td>88.94</td>
<td>87.08</td>
<td>88.82</td>
<td>82.72</td>
<td>84.94</td>
<td>77.46</td>
</tr>
<tr>
<td>DK_U</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>91.00</td>
<td>91.00</td>
<td>89.90</td>
<td>90.00</td>
<td>89.00</td>
<td>89.00</td>
<td>87.90</td>
</tr>
<tr>
<td>RE</td>
<td>89.90</td>
<td>89.90</td>
<td>85.90</td>
<td>84.20</td>
<td>84.50</td>
<td>84.00</td>
<td>62.00</td>
</tr>
<tr>
<td>NE</td>
<td>90.40</td>
<td>90.40</td>
<td>89.60</td>
<td>90.40</td>
<td>90.40</td>
<td>90.40</td>
<td>91.20</td>
</tr>
<tr>
<td>SE</td>
<td>89.98</td>
<td>89.99</td>
<td>86.76</td>
<td>89.89</td>
<td>89.28</td>
<td>89.90</td>
<td>84.01</td>
</tr>
<tr>
<td>DK_B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td>88.00</td>
<td>88.00</td>
<td>86.80</td>
<td>86.00</td>
<td>85.00</td>
<td>84.30</td>
<td>78.90</td>
</tr>
<tr>
<td>RE</td>
<td>85.40</td>
<td>85.20</td>
<td>84.90</td>
<td>78.40</td>
<td>71.20</td>
<td>73.90</td>
<td>48.80</td>
</tr>
<tr>
<td>NE</td>
<td>87.40</td>
<td>87.40</td>
<td>87.80</td>
<td>86.40</td>
<td>86.20</td>
<td>85.40</td>
<td>89.20</td>
</tr>
<tr>
<td>SE</td>
<td>87.98</td>
<td>87.97</td>
<td>86.73</td>
<td>87.89</td>
<td>86.64</td>
<td>84.33</td>
<td>73.97</td>
</tr>
</tbody>
</table>

**Results:** As observed in Section 6.4.2, USSVM beats SVM in terms of Robust Error here too. It indicates the fact that, USSVM is more Robust to uncertain kernels than that of SVM. However, for the other error measures, performance of USSVM is very similar to that of SVM. Although, ideally USSVM_{MDA} and USSVM_{FSS} should perform exactly same as USSVM_{SOCP}, but due to the approximate solution for the projection set the performance of USSVM_{FSS} is slightly different from that of USSVM_{SOCP}, while USSVM_{MDA} performs as good as USSVM_{SOCP}. Therefore, one can conclude that USSVM_{FSS} and USSVM_{MDA} give 100 times faster (see Figure 7.2) solution than that of USSVM_{SOCP} without compromising...
much in terms of accuracy. While in case of both \textbf{RSVM}$_{\text{MDA}}$ and \textbf{RSVM}$_{\text{FSS}}$. $\beta$ is approximated to positive-definite matrix instead of positive-semi-definite in case of \textbf{RSVM}$_{\text{SOCP}}$ and so sometimes performance of these two go a little down.

### 7.4.3 Verification of Convergence of FSS Algorithm

This section have experimentally verified that the proposed saddle point based algorithm has $O\left(\frac{1}{M_s^2}\right)$ convergence rate (see (7.43)). Recall that $M_s$ is the actual number of steps, which one can consider as iterations.

**Datasets:** The experiment was done on $\textbf{DKU}(1,N,5)$ with the value of $N \in \{50,500\}$ and $C \in \{1,10\}$.

**Results:** Figure 7.1 shows the convergence rate of the Saddle Point based algorithm for \textit{USSVM} and \textit{RSVM} formulation. The $x$-axis and the $y$-axis denote $\log_{10}(M_s)$ and $\log_{10}(\varepsilon_{sad})$ respectively. All the points on the graph indicate the “end” of the $s^{th}$ step and circled points indicate the “end” of the $s^*$ step. When $s > s^*$, ideally the graph should be a straight-line with slope less than $-2$ and one can observe the same in Figure 7.1. On the other hand, in case of \textit{USSVM} for $s < s^*$ rate of decrease in $\varepsilon_{sad}$ is much slower than the case in $s > s^*$ and opposite in case of \textit{RSVM}. This may be because of the fact that convergence of both the algorithms depend on all the different constants (for example $L_{xx}, L_{yy}$ etc) for $s < S^*$.

### 7.4.4 Comparison of Training Time

Along with verification of convergence criteria of proposed FSS algorithm it is interesting to compare the training time of FSS algorithm and MDA based procedure for both the formulations \textit{USSVM} and \textit{RSVM}.

As already mentioned, in case of the saddle point based algorithm, the number of stages and the number of iterations inside one stage do not depend on the number of data points (7.38). In each iteration of \textbf{USSVM}$_{\text{FSS}}$ one needs to solve two gradient projection type problems. It appears that they are extremely easy to compute, in one case there is a closed form solution, in
another case it could be solved by a line search algorithm. Similarly, in MDA based procedure the number of iterations does not depend on the number of data points. Therefore, both MDA based and FSS procedure solve the same problem solved by SOCP based algorithm, but the computation required may differ significantly. To this end, this experiment have compared the training time for $\text{USSVM}_{\text{SOCP}}$ with $\text{USSVM}_{\text{MDA}}$ and $\text{USSVM}_{\text{FSS}}$ and also $\text{RSVM}_{\text{SOCP}}$ with $\text{RSVM}_{\text{MDA}}$ and $\text{RSVM}_{\text{FSS}}$ with increasing the number of training data points ($n$). For this experiment, the following dataset has been used.

Figure 7.1: Rate of convergence for saddle point based algorithm for $\text{USSVM}$ (top) and $\text{RSVM}$ (bottom)
Datasets: The synthetic datasets were generated similar to $\text{DK}_U$ (please see Section 6.4.1) with number of data points in each class as \{250, 500, 1000, 1500, 2000, 2500\}, where $L = \{10, 50\}$. The values of $C = 10$ and $\kappa = 1$ were used. As training time of $\text{RSVM}$ does not depend on $L$, only the value $L = 10$ was used here.

Results: Figure 7.2 shows training time (in sec) for varying $N$. One can observe that, with the increase of $N$, the training time for $\text{USSVM}_{\text{SOCP}}$ and $\text{RSVM}_{\text{SOCP}}$ increases very steeply compared to the training time for FSS and MDA counterparts. As expected, training time for $\text{USSVM}$ in general increases with the increase of number of uncertain kernels. As an example, to build a robust classifier ($L = 50$) with only 3000 data points, $\text{USSVM}_{\text{SOCP}}$ needs more than 5 hours while $\text{USSVM}_{\text{FSS}}$ completes within 40 minutes and $\text{USSVM}_{\text{MDA}}$ within 30 minutes. Though convergence rate of FSS is higher than that of MDA, but that is for more accurate region (beyond the range of $e^{-10}$). In these set of experiments, both the algorithms continue only up to convergence in order of $e^{-8}$. This may be a reason behind less training time for MDA based procedure than that of FSS algorithm. This concludes that, to build a robust classifier
with a medium scale of data (even more than 1000) the saddle point based or MDA based algorithm is much more effective as the Quadratic Conic Program based formulation is considerably slow and it becomes expensive to build robust classifiers. Of course FSS procedure is better if one need accuracy in order of $e^{-10}$ or more.

**Discussion of experimental results** The results on the synthetic experiments show that $\text{USSVM}_{\text{MDA}}$, $\text{USSVM}_{\text{FSS}}$ perform similar to $\text{USSVM}_{\text{SOCP}}$ and also the performance of $\text{RSVM}_{\text{MDA}}$, $\text{RSVM}_{\text{FSS}}$ slightly decreases but very close to that of $\text{RSVM}_{\text{SOCP}}$ in terms of generalization as measured by various error measures. It is also demonstrated that FSS and MDA based algorithms are much more scalable than SOCP based one. Even to build a robust classifier with 3000 data points, $\text{USSVM}_{\text{SOCP}}$ needs more than 5 hours while $\text{USSVM}_{\text{FSS}}$ completes within 40 minutes.

### 7.5 Summary

This chapter studied efficient solver for the problem of designing classifiers when the Kernel matrix is uncertain. It is applied for both the *additive noise* and the *affine uncertainty set* setting. Mirror descent based algorithms are proposed for both with convergence $O(1/\sqrt{T})$. Also a novel FSS procedure is used to solve the problem efficiently which has $O(1/T^2)$ convergence. Empirical results further justify the claims.
Chapter 8

Conclusion

Abstract

This final chapter summarizes the main contributions and discusses related issues, open problems and possible directions for future work.

8.1 Summary

The Thesis has presented the idea of building Robust Classifiers for handling interval valued uncertainty in observations presented by feature vectors or kernel matrix, leading to non-overly conservative linear and nonlinear robust classifiers with better generalization. A novel maximum-margin linear classifier has been proposed which achieves good generalization on data having interval-valued uncertainty. The key idea has been to employ chance-constraints in order to handle uncertainty. The main contribution is to derive a tractable SOCP formulation, which is a safe approximation of the resulting CCP, using Bernstein schemes. The formulation ensures that the probability of misclassification on interval data is less than a user specified upper-bound ($\varepsilon$). Also, the geometric interpretation shows that the proposed classifiers interpret each training example as the region of intersection of its bounding hyper-rectangle and an ellipsoid centered at its mean.

Moreover, a novel methodology for constructing robust classifiers by employing partial
information of support and moments of the uncertain training datapoints have also been presented here. For this setup too, the idea was to pose the uncertain data classification problem as a CCP and relax it as a convex SOCP formulation using Bernstein bounding schemes. The key advantage of the Bernstein relaxation scheme is to model uncertainty in a less conservative manner. Since the relaxation requires the knowledge of bounds on moments rather than the exact moments themselves, the resulting classifiers are also inherently robust to moment estimation errors. Using the proposed methodology, various robust formulations employing different levels of partial information have been derived. Interesting error measures for evaluating performance of classifiers robust to uncertain data have also been presented.

The performance of the proposed classifiers were empirically evaluated on various synthetic and real-world datasets. Experiment results show that in general, the Bernstein relaxation schemes are less conservative than the Chebyshev based schemes. Hence the proposed linear classifiers are robust to interval-valued uncertainty and also not-overly-conservative. They achieve higher margins and hence better generalization than state-of-the-art. Moreover, formulations using bounds on moments not only achieves good generalization but are also less susceptible to moment estimation errors.

This Thesis has also presented an optimization problem of building non-linear classifier RSVM which is robust to uncertainty in the kernel matrix. The formulation applies to Gaussian uncertainty as well as to arbitrary distributions with finite support. The idea here also was to pose the problem as Chance-Constraint-Program and then to approximate with the help of Bernstein schemes. In the finite support case, the formulation has been derived from a novel large deviation inequality using Bernstein scheme, stated in Theorem 5.2. The large deviation inequality is of independent interest and applies more generally to problems involving traces of random matrices. An interesting result is that, for i.i.d uncertainty the formulation reduces to SVM (Theorem 5.5). Solving co-positive programs are extremely difficult but for positive semidefinite $\beta$ the formulation is second-order-cone representable and can be solved by SOCP. The results on the synthetic experiments proved that the proposed RSVM performs better than Nominal SVM in terms of generalization as measured by various error measures and also more robust than the Nominal SVM. Moreover, the formulation when applied to protein structure
classification yields significantly improved results.

But this CCP based approaches was unable to capture the information that every uncertain kernel should be a valid kernel. Hence modeling the uncertainty in kernel were inappropriate here. Therefore, the Thesis has also tried to solve the same problem of designing classifiers when the kernel matrix is uncertain but considering the fact that any uncertain kernel is a valid one. It defined an affine uncertainty set of uncertain kernels which leads to a valid model of uncertainty over kernel matrices. Using a robust optimization based framework, the problem of designing the robust classifier was posed as a minimax problem. The minimax procedure is solved as a second-order-conic convex program. The proposed robust classifier was called as \textbf{USSVM}. The results on the synthetic experiments show that \textbf{USSVM} performs better than \textbf{RSVM} in terms of generalization as measured by various error measures. Proposed formulation \textbf{USSVM} is more robust than the non-robust \textit{Nominal SVM}. The formulation when applied to protein structure data yields significantly improved results.

The Thesis has also proposed efficient solvers for the problem of designing classifiers when the kernel matrix is uncertain. Mirror descent based algorithms (MDA) has been proposed with the order of convergence as $O(1/\sqrt{T})$, where $T$ is number of iterations. Also a novel FSS procedure has been proposed and used to solve the problem efficiently with the order of convergence as $O(1/T^2)$. Both of them are applied to the additive noise and the affine uncertainty set setting. The results on the synthetic experiments show that FSS and MDA based algorithms are much more scalable than IP method based SOCP solvers.

\subsection*{8.2 Future Directions}

- The Bernstein relaxation schemes presented in this Thesis are not only aid in approximating the original CCP as a convex program, but also open avenues for efficient approximations of other CCP-based learning formulations (e.g. [8] and its variants). By employing rich partial information, the Bernstein schemes lead to less conservative relaxations. Hence exploitation of the Bernstein schemes in the context of learning is a good direction for research.
• In future, it would be interesting to extend this Maximum-Margin methodology (proposed IC-SM or MM-SM-I) for applying in to extended feature spaces using the kernel trick. Note that, all the proposed formulations involve the function \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) and the matrix, \( L_i \), computed using the support information. Hence it is not straight forward to kernelize the formulations unless approaches like constraint sampling [47] are employed. However the geometric interpretation of the proposed formulations may suggest an easy way to kernelize some of the proposed formulations. It would be very interesting to explore the applicability of the proposed methods to the dual space using arbitrary kernel function.

• The large deviation inequality stated in Theorem 5.2, is of independent interest and can be applied to more general problems involving traces of random matrices. Exploring other learning problems dealing with random matrices using this inequality can be an interesting direction of research.

• Extending the CCP based formulation to symmetric property of kernel is straightforward. It would be interesting to derive a CCP based model including the positive-semidefinite properties of a kernel matrix.

• It will be interesting to explore the applicability of Bernstein scheme or the proposed The large deviation inequality stated in Theorem 5.2 to the robust variant of other learning algorithms like logistic classification and regression.
Appendix A

Robust Optimization

A.1 Introduction to Robust Optimization

This section provides a brief background of robust optimization and refers the reader to the survey [22], the textbook [23] and references to the original papers therein, for more details.

Robust Linear Programming

Consider uncertain LP as:

$$\begin{align*}
\min_x & \quad c^T x \\
\text{s.t.} & \quad a_i^T x \leq b_i, \quad \forall i = 1, \ldots, m. \\
& \quad x \in \mathbb{R}^n
\end{align*}$$

(A.1)

where, $c, a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$ and $a_i$ is uncertain vector. Please note that, for existence of tractable robust counter part of above uncertain problem all constraints containing uncertain parameter must be inequality constraints.

Result A.1. If uncertainty is of box type, i.e., $a_i \in \{a_i \mid D_i a_i \leq d_i\}$ then the robust counter
parts turns out to be:

\[
\min_x \ c^\top x \\
\text{s.t. } \ p_i^\top d_i \leq b_i, \ \forall i = 1, \ldots, m. \\
\ p_i^\top D_i = x, \ \forall i = 1, \ldots, m. \\
\ p_i \geq 0, \ \forall i = 1, \ldots, m. \\
\ x \in \mathbb{R}^n
\]  

(A.2)

Note that, it is an LP itself [22].

**Result A.2.** If uncertainty set represented by an ellipsoidal region \( \mathbf{a}_i = \mathbf{a}_i^0 + \mathbf{u}_i \), where \( \|\mathbf{u}_i\|_p \leq \rho_i \) then the robust counter part of uncertain LP(A.1) is as follows:

\[
\min_x \ c^\top x \\
\text{s.t. } \ a_i^0^\top x + \rho_i \|x\|_q \leq b_i, \ \frac{1}{p} + \frac{1}{q} = 1, \ \forall i = 1, \ldots, m. \\
\ x \in \mathbb{R}^n
\]  

(A.3)

This is an convex conic program. For \( p = 2 \), this turns out to be a second-order-cone-program(SOCP). For more details please refer [55].

**Robust Quadratic Problem** Consider an uncertain convex quadratically constrained problem:

\[
\min_x \ c^\top x \\
\text{s.t. } \ x^\top A x \leq 2b^\top x + d, \ (A, b, d) \in \mathcal{U} \\
\ x \in \mathbb{R}^n
\]  

(A.4)

Here, in contrast to the case of LP, uncertainty sets even of fairly simple geometry (e.g., a
box or a polyhedral) can yield NP-hard and thus computationally intractable robust counterparts [23]. In these cases, the Robust Optimization methodology recommends solution using an approximate robust counterpart instead of the true one. When uncertain set $\mathcal{U}$ is ellipsoidal, it can be represented as $\mathcal{U} = \{(A, b, d) := (A^0, b^0, d^0) + \sum_{i=1}^{L} u_i(A^i, b^i, d^i) : \|u\|_2 \leq 1\}$. Then the robust counterpart turns out to be a semi-definite program (SDP):

$$
\begin{align*}
\min_x & \quad c^T x \\
\text{s.t.} & \quad M \succeq 0 \\
& \quad x \in \mathbb{R}^n
\end{align*}
$$

(A.5)

where

$$
M = \begin{pmatrix}
\begin{array}{cccc}
d^0 + 2x^T b^0 - \tau & \frac{1}{2}d^1 + x^T b^1 & \ldots & \frac{1}{2}d^L + x^T b^L \\
\frac{1}{2}d^1 + x^T b^1 & \tau & \ddots & (A^1x)^T \\
\vdots & \ddots & \ddots & \vdots \\
\frac{1}{2}d^L + x^T b^L & \ldots & \tau & (A^Lx)^T \\
A^0x & A^1x & \ldots & A^Lx & I
\end{array}
\end{pmatrix}
$$

Therefore, one can conclude that under ellipsoidal uncertainty sets, the robust counterpart of an LP becomes an SOCP and an SOCP becomes an SDP. However, the robust counterpart of an SDP is NP-hard to solve (please see [23]). In other words, the difficulty of the robust problem increases, as SDPs are numerically harder to solve than SOCPs, which in turn are harder to solve than LPs. Hence, a practical drawback of such an approach is that, it leads to nonlinear, although convex models, which are computationally more complex.
Appendix B

Numerical Computations

B.1 Computation of $v(\hat{\mu})$

This section presents details of the numerical procedure for computing $v(\hat{\mu})$. Consider the following claim:

**Claim B.1.** Let $v(\hat{\mu})$ be as defined in (3.8). Then, $\sqrt{1-\hat{\mu}^2} \leq v(\hat{\mu}) \leq 1$.

**Proof.** Recalling the definition of $v(\hat{\mu})$:

$$v(\hat{\mu}) = \min \{ v \geq 0 \mid f(t; v, \hat{\mu}) \geq 0, \forall t \in \mathbb{R} \}$$

where $f(t; v, \hat{\mu}) \equiv \frac{v^2}{2} t^2 + \hat{\mu} t - \log (\cosh(t) + \hat{\mu} \sinh(t))$. Let $f'(t; v, \hat{\mu}) = g_1(t) - g_2(t)$ where $g_1(t) \equiv v^2 t + \hat{\mu}$ and $g_2(t) \equiv \frac{\sinh(t) + \hat{\mu} \cosh(t)}{\cosh(t) + \hat{\mu} \sinh(t)}$. Now, if $g_1'(0) < g_2'(0)$, then there exists a neighbourhood around $t = 0$ where $f'(t) < 0$ (since $f'(0) = 0$). Also in this neighbourhood $f(t) < 0$ because $f(0) = 0$. Thus $g_1'(0) \geq g_2'(0)$ is a necessary condition for $f \geq 0$. In other words, $v(\hat{\mu}) \geq \sqrt{1-\hat{\mu}^2}$. Also, from proof of Lemma 3.2, one gets $v(\hat{\mu}) \leq 1$. This completes the proof. \qed

Note that, the function $f$ strictly increases with the value of $v$ and by Claim B.1 one gets $\sqrt{1-\hat{\mu}^2} \leq v(\hat{\mu}) \leq 1$. Thus one can have a simple binary search algorithm for computing $v$. The algorithm starts with $v_0' \equiv \sqrt{1-\hat{\mu}^2}$ and $v_0'' \equiv 1$. At every iteration, $i \geq 1$, $v_i \equiv \frac{v_{i-1}'+v_{i-1}''}{2}$.
Appendix B. Numerical Computations

<table>
<thead>
<tr>
<th>Table B.1: Table showing values of ( \nu ) as a function of ( \hat{\mu} \in [0, 1] ) at 20 equal increments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
</tr>
<tr>
<td>0.9481</td>
</tr>
</tbody>
</table>

and it is checked whether \( f^{\min}_i = \min_t f(t; \nu_i, \hat{\mu}) \geq 0 \). If \( f^{\min}_i \leq 0 \), then \( \nu_i^l = \nu_i \), else \( \nu_i^l = \nu_i \).

This is repeated until a relevant stopping criteria is met. Also, as the proof of Claim B.1 suggests, for fixed values of \( \nu, \hat{\mu} \), the function \( f \) has only one minimum wrt. \( t \) (this is because \( g_2(t) \) is concave above \( t = t^* \) and convex below \( t = t^* \)). Hence checking whether \( f^{\min}_i \geq 0 \) for a fixed value \( \nu_i \) is also easy. The values of \( \nu \) as a function of \( \hat{\mu} \in [0, 1] \) are shown in table B.1 and also in and figure 3.2. Since the function \( \nu(\hat{\mu}) \) is symmetric wrt. \( \hat{\mu} \), then \( \nu(\hat{\mu}) = \nu(-\hat{\mu}) \).

### B.2 Computation of \( \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \)

This section presents details of the numerical procedure for computing \( \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) (refer (C.4)). Recall that \( |\hat{\mu}^| \leq \hat{\sigma} \leq 1 \). Now consider the following claim:

**Claim B.2.** Let \( \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) be as defined in (C.4). Then, \( \sqrt{\hat{\sigma}^2 - (\hat{\mu}^{\min})^2} \leq \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \leq 1 \), where \( \hat{\mu}^{\min} = \min(|\hat{\mu}^-|, |\hat{\mu}^+|) \).

**Proof.** Rewriting the definition of \( \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \):

\[
\nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) = \min \left\{ c \geq 0 \mid f(t; \hat{\mu}, \hat{\sigma}, c) \geq 0, \forall t \in \mathbb{R}, \forall \hat{\mu} \in [\hat{\mu}^-, \hat{\mu}^+] \right\}
\]

where \( f(t; \hat{\mu}, \hat{\sigma}, c) \equiv \frac{c^2}{2} t^2 + \max[\hat{\mu}^- t, \hat{\mu}^+ t] - h_{\hat{\mu}, \hat{\sigma}}(t) \) (refer (C.2) for definition of \( h_{\hat{\mu}, \hat{\sigma}}(t) \)).

Now, let \( t \geq 0 \) and \( f'(t; \hat{\mu}, \hat{\sigma}, c) = g_1(t) - g_2(t) \) where \( g_1(t) \equiv \hat{\sigma}^2 t + \hat{\mu}^+ \) and

\[
g_2(t) \equiv \frac{(1 - \hat{\mu}) (\hat{\mu} - \hat{\sigma}^2) \exp \left\{ t \frac{\hat{\mu} - \hat{\sigma}^2}{1 - \hat{\mu}} \right\} + (\hat{\sigma}^2 - \hat{\mu}^2) \exp \left\{ t \right\}}{(1 - \hat{\mu}) \exp \left\{ t \frac{\hat{\mu} - \hat{\sigma}^2}{1 - \hat{\mu}} \right\} + (\hat{\sigma}^2 - \hat{\mu}^2) \exp \left\{ t \right\}}
\]

Now, if \( g_1'(0) < g_2'(0) \), then there exists a neighbourhood around \( t = 0 \) where \( f'(t; \hat{\mu}^+, \hat{\sigma}) < 0 \) (since \( f'(0; \hat{\mu}^+, \hat{\sigma}) = 0 \)). Also in this neighbourhood \( f(t; \hat{\mu}^+, \hat{\sigma}) < 0 \) because \( f(0; \hat{\mu}^+, \hat{\sigma}) = 0 \). Thus \( g_1'(0) \geq g_2'(0) \) is a necessary condition for \( f \geq 0 \). In other words, \( \nu(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \geq \ldots \)
Appendix B. Numerical Computations

\[ \sqrt{\bar{s}^2 - (\hat{\mu}^+)^2}. \] A similar analysis for \( t \leq 0 \) gives \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \geq \sqrt{\bar{s}^2 - (\hat{\mu}^-)^2}. \) Also, from the very definition of \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \), one gets its value \( \leq 1 \) (refer Lemma 4.1). This proves the claim. \( \Box \)

Note that, the function \( f(t; \hat{\mu}, \hat{\sigma}, c) \) strictly increases with the value of \( c \) and by the above claim one can get \( \sqrt{\bar{s}^2 - (\hat{\mu}_{\text{min}})^2} \leq c \leq 1. \) Thus one can have a simple binary search algorithm for computing \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \). The algorithm starts with \( c_0^t \equiv \sqrt{\bar{s}^2 - (\hat{\mu}_{\text{min}})^2} \) and \( c_0^t \equiv 1. \) At every iteration, \( i \geq 1, c_i \equiv \frac{c_{i-1}^t + c_{i-1}^n}{2} \) and it is checked whether \( f_i^\text{min} \equiv (\min_t f(t; \hat{\mu}, \hat{\sigma}, c_i)) \geq 0. \) If \( f_i^\text{min} \geq 0, \) then \( c_i^t \equiv c_i, \) else \( c_i^t \equiv c_i. \) This is repeated until a relevant stopping criteria is met. Checking whether \( f_i^\text{min} \geq 0 \) for a fixed value \( c_i \) can be done using any 1-d minimization routine. Table B.2 shows values of \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) computed using this numerical procedure. For each value of \( \hat{\sigma}, \) \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) is computed for 10 equally spaced \( \hat{\mu}^\pm \) values in the range \([-\hat{\sigma}, \hat{\sigma}]. \) In the table, \( \hat{\mu}^- \) and \( \hat{\mu}^+ \) vary across rows and columns respectively. Hence a ‘–’ represents the case \( \hat{\mu}^- > \hat{\mu}^+ \) (which is not allowed).

Table B.2: Table showing values of \( v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \) computed using the numerical procedure.

| \( \hat{\sigma} = 0.2 \) |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| 0.0000          | 0.3770          | 0.3875          | 0.3960          | 0.3922          | 0.3796          | 0.3580          | 0.3368          | 0.3233          | 0.3090          |
| –               | 0.3770          | 0.3875          | 0.3960          | 0.3922          | 0.3796          | 0.3580          | 0.3367          | 0.3232          | 0.3233          |
| –               | -1.0000         | 0.3875          | 0.3960          | 0.3922          | 0.3796          | 0.3580          | 0.3368          | 0.3367          | 0.3368          |
| –               | –               | –               | 0.3960          | 0.3922          | 0.3796          | 0.3582          | 0.3580          | 0.3580          | 0.3580          |
| –               | –               | –               | –               | 0.3922          | 0.3796          | 0.3796          | 0.3796          | 0.3796          | 0.3796          |
| –               | –               | –               | –               | –               | 0.3922          | 0.3922          | 0.3922          | 0.3922          | 0.3922          |
| –               | –               | –               | –               | –               | –               | 0.3960          | 0.3960          | 0.3960          | 0.3960          |
| –               | –               | –               | –               | –               | –               | –               | 0.3875          | 0.3875          | 0.3875          |
| –               | –               | –               | –               | –               | –               | –               | –               | 0.3770          | 0.3770          |
| –               | –               | –               | –               | –               | –               | –               | –               | –               | 0.0000          |

continued on next page
Table B.2 —continued from previous page

<table>
<thead>
<tr>
<th>$\hat{s} = 0.4$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
<tr>
<td>–</td>
<td>0.5096</td>
</tr>
</tbody>
</table>
Table B.2 —continued from previous page

<table>
<thead>
<tr>
<th></th>
<th>0.7893</th>
<th>0.8224</th>
<th>0.8206</th>
<th>0.7999</th>
<th>0.7543</th>
<th>0.6652</th>
<th>0.6652</th>
<th>0.6652</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>0.8224</td>
<td>0.8206</td>
<td>0.7999</td>
<td>0.7543</td>
<td>0.7543</td>
<td>0.7543</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.8206</td>
<td>0.7999</td>
<td>0.7999</td>
<td>0.7999</td>
<td>0.7999</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.8206</td>
<td>0.8206</td>
<td>0.8206</td>
<td>0.8206</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.8224</td>
<td>0.8224</td>
<td>0.8224</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.7893</td>
<td>0.7893</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.7454</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

\[ \hat{s} = 1 \]

<table>
<thead>
<tr>
<th></th>
<th>0.0000</th>
<th>0.8579</th>
<th>0.9418</th>
<th>0.9804</th>
<th>0.9976</th>
<th>0.9938</th>
<th>0.9428</th>
<th>0.8315</th>
<th>0.6285</th>
<th>0.0000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>–</td>
<td>0.8579</td>
<td>0.9418</td>
<td>0.9804</td>
<td>0.9976</td>
<td>0.9938</td>
<td>0.9428</td>
<td>0.8315</td>
<td>0.6285</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>0.9418</td>
<td>0.9804</td>
<td>0.9976</td>
<td>0.9938</td>
<td>0.9428</td>
<td>0.8315</td>
<td>0.8315</td>
<td>0.8315</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.9804</td>
<td>0.9976</td>
<td>0.9938</td>
<td>0.9428</td>
<td>0.9428</td>
<td>0.9428</td>
<td>0.9428</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.9976</td>
<td>0.9938</td>
<td>0.9938</td>
<td>0.9938</td>
<td>0.9938</td>
<td>0.9938</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.9976</td>
<td>0.9976</td>
<td>0.9976</td>
<td>0.9976</td>
<td>0.9976</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.9804</td>
<td>0.9804</td>
<td>0.9804</td>
<td>0.9804</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.9418</td>
<td>0.9418</td>
<td>0.9418</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.8579</td>
<td>0.8579</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix C

Proof of Lemmas

C.1 Proof of Lemma 4.1

This section presents the proof of Lemma 4.1:

Proof. Consider the random variable \( \hat{x}_{ij} = \frac{x_{ij} - m_{ij}}{l_{ij}} \). It is easy to see that \(-1 \leq \hat{x}_{ij} \leq 1\) and \(E[\hat{x}_{ij}] = \hat{\mu}_{ij}, E[\hat{x}_{ij}^2] = \hat{\sigma}_{ij}^2\). By Jensen’s inequality, one gets \(|\hat{\mu}_{ij}| \leq \hat{\sigma}_{ij}\). Hence, without loss of generality, assume \(|\hat{\mu}_{ij}| \leq \hat{\sigma}_{ij}\). Now, \(E[\exp\{tx_{ij}\}] = E[\exp\{tl_{ij}\hat{x}_{ij}\}] + \exp\{tm_{ij}\}\). Let \(\tilde{t} = tl_{ij} \). According to Table 2 in [10]:

\[
E[\exp\{t\hat{x}_{ij}\}] \leq g_{\hat{\mu}_{ij}, \hat{\sigma}_{ij}}(\tilde{t}) = \left\{ \begin{array}{ll}
\frac{(1-\hat{\mu}_{ij})^2 \exp\left\{\frac{\hat{\mu}_{ij} - \hat{\sigma}_{ij}^2}{1-\hat{\mu}_{ij}}\right\} + \left(\hat{\sigma}_{ij}^2 - \hat{\mu}_{ij}^2\right) \exp(\tilde{t})}{1-2\hat{\mu}_{ij} + \hat{\sigma}_{ij}^2}, & \tilde{t} \geq 0 \\
\frac{(1+\hat{\mu}_{ij})^2 \exp\left\{\frac{\hat{\mu}_{ij} + \hat{\sigma}_{ij}^2}{1+\hat{\mu}_{ij}}\right\} + \left(\hat{\sigma}_{ij}^2 - \hat{\mu}_{ij}^2\right) \exp(-\tilde{t})}{1+2\hat{\mu}_{ij} + \hat{\sigma}_{ij}^2}, & \tilde{t} \leq 0
\end{array} \right.
\] (C.1)

Now consider the function

\[h_{\hat{\mu}_{ij}, \hat{\sigma}_{ij}}(\tilde{t}) \equiv \log g_{\hat{\mu}_{ij}, \hat{\sigma}_{ij}}(\tilde{t})\] (C.2)
It is easy to show that $h_{\hat{\mu}, \sigma_j}(0) = 0$ and $h_{\hat{\mu}, \sigma_j}(0) = \hat{\mu}_j$. Now for $\tilde{i} \geq 0$,

$$h''_{\hat{\mu}, \sigma_j}(\tilde{i}) = \frac{(\hat{\sigma}_j^2 - \hat{\mu}_j^2)(1 - 2\hat{\mu}_j + \hat{\sigma}_j^2)^2 \exp\left\{\tilde{i} \frac{\hat{\mu}_j - \hat{\sigma}_j^2}{1 - \hat{\mu}_j} \right\} \exp\{\tilde{i}\}}{(1 - \hat{\mu}_j)^2 \exp\left\{\tilde{i} \frac{\hat{\mu}_j - \hat{\sigma}_j^2}{1 - \hat{\mu}_j} \right\} + (\hat{\sigma}_j^2 - \hat{\mu}_j^2) \exp\{\tilde{i}\}} \frac{4 (\hat{\sigma}_j^2 - \hat{\mu}_j^2)(1 - \hat{\mu}_j)^2 \exp\left\{\tilde{i} \frac{\hat{\mu}_j - \hat{\sigma}_j^2}{1 - \hat{\mu}_j} \right\} \exp\{\tilde{i}\}}{(1 - \hat{\mu}_j)^2 \exp\left\{\tilde{i} \frac{\hat{\mu}_j - \hat{\sigma}_j^2}{1 - \hat{\mu}_j} \right\} + (\hat{\sigma}_j^2 - \hat{\mu}_j^2) \exp\{\tilde{i}\}} \leq 1 \quad (C.3)$$

The last inequality is true by the AM-GM inequality. Similarly one can derive an inequality for the case $\tilde{i} \leq 0$. Thus $h''_{\hat{\mu}, \sigma_j}(\tilde{i}) \leq 1 \forall \tilde{i}$. Using Taylor series, it follows that $h_{\hat{\mu}, \sigma_j}(\tilde{i}) \leq \hat{\mu}_j \tilde{i} + \frac{1}{2} \hat{\sigma}_j^2 \tilde{i}^2$. As a result, the function:

$$v(\hat{\mu}^-, \hat{\mu}^+, \hat{\sigma}) \equiv \min \left\{ c \geq 0 : h_{\hat{\mu}, \sigma}(\tilde{i}) \leq \max[\hat{\mu}^- \tilde{i}, \hat{\mu}^+ \tilde{i}] + \frac{c^2}{2} \tilde{i}^2 \forall (\hat{\mu} \in [\hat{\mu}^-, \hat{\mu}^+], \tilde{i}) \right\} \quad (C.4)$$

is well defined ($0 \leq v(\ldots) \leq 1$) and hence $g_{\hat{\mu}, \sigma_j}(\tilde{i}) \leq \exp\left\{ \max[\hat{\mu}_j \tilde{i}, \hat{\mu}_j^+ \tilde{i}] + \frac{v(\hat{\mu}_j^-, \hat{\mu}_j^+, \hat{\sigma})^2}{2} \tilde{i} \right\} \forall \tilde{i}$. Noting that $g_{\hat{\mu}, \sigma_j}(\tilde{i})$ is an upper bound on $E \left[ \exp\{\tilde{i} \hat{x}_{ij}\} \right]$ and using the fact that $\mu_j^\pm = l_{ij} \hat{\mu}_j^\pm + m_{ij}$, (4.9) is obtained. This completes the proof of Lemma 4.1.

**C.2 Proof of Lemma 4.4**

This section presents the proof of Lemma 4.4:

*Proof.* Recall the definition of the function $h_{\hat{\mu}, \sigma}(\tilde{i})$ from (C.1,C.2). It is easy to see that $h_{\hat{\mu}, 1}(\tilde{i}) = \log(\cosh \tilde{i} + \hat{\mu} \sinh \tilde{i}), \tilde{i} \in \mathbb{R}$. Thus $\nu(\hat{\mu}^-, \hat{\mu}^+, 1)$ (defined in (C.4)) is the minimum value of $c$ for which $f(\tilde{i}) \geq 0, \forall \tilde{i}$, where $f(\tilde{i})$ is defined as follows:

$$f(\tilde{i}) \equiv \begin{cases} \frac{c_1^2}{2} \tilde{i}^2 + \hat{\mu}^+ \tilde{i} - \log(\cosh \tilde{i} + \hat{\mu}^+ \sinh \tilde{i}), & \tilde{i} \geq 0 \\ \frac{c_2^2}{2} \tilde{i}^2 + \hat{\mu}^- \tilde{i} - \log(\cosh \tilde{i} + \hat{\mu}^- \sinh \tilde{i}), & \tilde{i} < 0 \end{cases}$$

Now, consider the case, $t \geq 0$. Let $f'(\tilde{i}) = g_1(\tilde{i}) - g_2(\tilde{i})$ where $g_1(\tilde{i}) \equiv c_1^2 \tilde{i}^2 + \hat{\mu}^+, g_2(\tilde{i}) \equiv \frac{c_1^2}{2} \tilde{i}^2 + \hat{\mu}^- \tilde{i} - \log(\cosh \tilde{i} + \hat{\mu}^- \sinh \tilde{i})$. The proof follows from this.
Now the following claim is true:

**Claim C.1.** \( g_2(\tilde{t}) \) is concave for \( \tilde{t} \geq 0 \).

**Proof.** The value of \( g''(\tilde{t}) \) can be calculated as \( \frac{8(1-\tilde{\mu}^+)^2}{(1+\tilde{\mu}^+)^2 - (1+\tilde{\mu}^-)^2} \). Also \( g''(\tilde{t}) \leq 0 \iff \tilde{t} \geq \frac{1}{2} \log \left( \frac{1-\tilde{\mu}^+}{1+\tilde{\mu}^+} \right) \). This says that \( g_2(\tilde{t}) \) is concave for \( t \geq 0 \), proving the claim. \( \square \)

Since \( g_2(\tilde{t}) \) is concave, \( g_1'(0) \geq g_2'(0) \) implies \( f'(\tilde{t}) \geq 0, \forall \tilde{t} \geq 0 \) and is thus a sufficient condition for \( f(\tilde{t}) \geq 0, \forall \tilde{t} \geq 0 \), as \( f(0) = 0 \). Also if \( g_1'(0) < g_2'(0) \) then \( \exists \) a neighbourhood in \( \tilde{t} \geq 0 \) where \( f'(\tilde{t}) < 0 \) which implies \( f(\tilde{t}) < 0 \) in that neighbourhood since \( f(0) = 0 \). Thus \( g_1'(0) \geq g_2'(0) \) is a necessary and sufficient condition for \( f(\tilde{t}) \geq 0, \forall \tilde{t} \geq 0 \). In other words \( c^2 \geq 1 - (\tilde{\mu}^+)^2 \).

Similar arguments for \( \tilde{t} < 0 \) give the condition \( c^2 \geq 1 - (\tilde{\mu}^-)^2 \). Defining \( \tilde{\mu}_{\text{min}} = \min(|\tilde{\mu}^-|, |\tilde{\mu}^+|) \), one gets \( v(\tilde{\mu}^-, \tilde{\mu}^+, 1) = \sqrt{1 - (\tilde{\mu}_{\text{min}})^2} \). This completes the proof. \( \square \)
Appendix D

Proofs for the Saddle Point Algorithm

D.1 Proof of Lemma 7.2

Since $\omega_s(\cdot)$ is a d.-g.f. for $\mathcal{E}_v$, this function is convex and continuously differentiable on the entire $\mathcal{E}_v$. It follows that $\omega_s(x,v)$ is continuous convex function on $\mathcal{E}$, the set $\mathcal{E}^0 := \{ (x,v) : \partial \omega_s(z) \neq \emptyset \}$ is equal to $\{x : \partial \omega_s(x) \neq \emptyset \} \times \mathcal{V}$, and $\omega_s(x,v)$ admits a continuous on $\mathcal{E}^0$ selection of subgradient. All is equired to complete the verification of the fact that $\omega_s$ is a d.-g.f.

It remains to prove (7.36). Relation (7.36.a) is evident. To verify (7.36.b), let $z = (x,v), z' = (x',v') \in \mathcal{E}^0$, let $\Delta x = x' - x$, $\Delta v = v' - v$, $\xi = \|\Delta x\|_x \sqrt{\alpha_s/\Omega_x}$, $\eta = \|\Delta v\|_v \sqrt{\beta_s/\Omega_v R_s^{-1}}$, so that
Appendix D. Proofs for the Saddle Point Algorithm

\[ \|z - z'\|_{(s)} = \|[[\xi; \eta]]\|_2 \text{ by (7.35)} \]

\[ \|G_x(z) - G_x(z')\|_x \leq L_{xx}\|v - v'\|_v, \|G_v(z) - G_v(z')\|_v \leq L_{vv}\|v - v'\|_v \]

[see (7.34)]
\[ \Rightarrow \|G(z) - G(z')\|_{(s),*}^2 \leq \frac{\Omega_x \Omega_v}{\alpha_s \beta_s} L_{xx}^2 \|\Delta v\|_v^2 + \frac{\Omega_x R_s^2}{\beta_s} \left[ L_{xx}\|\Delta x\|_x + L_{vv}\|\Delta v\|_v \right]^2 \]
\[ = \frac{\Omega_x \Omega_v^2 i^2}{\alpha_s \beta_s} \eta^2 + \frac{\Omega_v R_s^2}{\beta_s} \left[ L_{xx} \sqrt{\Omega_x / \alpha_s} \xi + L_{vv} R_s \sqrt{\Omega_v / \beta_s} \eta \right]^2 \]
\[ = \|M[\xi; \eta]\|_2^2, \]
\[ M = \begin{bmatrix} \sqrt{\frac{\Omega_x \Omega_v}{\alpha_s \beta_s} R_s L_{xx}} \\ \sqrt{\frac{\Omega_v R_s^2}{\beta_s} L_{vv}} \end{bmatrix} = L_{sN}, \quad N = \begin{bmatrix} \sqrt{(1 - \beta_s) / \beta_s} \\ \sqrt{2 \beta_s - 1) / \beta_s} \end{bmatrix}, \]

where the relations in the last line are readily given by (7.35). In view of this computation and the fact that \( \|z - z'\|_{(s)} \leq \|[[\xi; \eta]]\|_2 \), in order to verify (7.36.b) it suffices to show that the spectral norm of the symmetric matrix \( N \) is \( \leq 1 \); since \( N \) is nonnegative due to \( \beta_s \geq 1/2 \), see (7.35), the latter task is exactly the same as verifying positive semidefiniteness of the matrix \( I_2 - N \), which is immediate.

D.2 Proof of Proposition 7.1

Consider stage \( s \), assuming that \( (I_s) \) take place.

1°. For \( z \in X^0 \) and \( \xi \in \mathcal{E} := \mathcal{E}_x \times \mathcal{E}_v \), let

\[ \text{Prox}_z(\xi) = \arg\min \{ \langle \xi - \omega_s(z), u \rangle + \omega_s(u) \} \]

the right-hand side is well defined due to strong convexity of \( \omega_s(\cdot) \). The basic observation is as follows:
Lemma D.1. [45, cf. Lemma 3.1] Given \( z \in \mathcal{X}^0 \), \( \xi, \eta \in \mathcal{E} \), let \( w = \text{Prox}_z(\xi) \) and \( z_+ = \text{Prox}_z(\eta) \). Then for all \( u \in \mathcal{X} \) it holds

\[
\langle \eta, w - u \rangle \leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta, w - z_+ \rangle - V_z^s(z_+) \tag{a}
\]

\[
\leq V_z^s(u) - V_{z_+}^s(u) + \langle \eta - \xi, w - z_+ \rangle - V_z^s(w) - V_{z_+}^s(z_+) \tag{b}
\]

\[
\leq V_z^s(u) - V_{z_+}^s(u) + \left[ \frac{1}{2} \| \eta - \xi \|_{(s),*} \| w - z_+ \|_{(s)} - \frac{1}{2} \| z - w \|_{(s)}^2 - \frac{1}{2} \| z_+ - w \|_{(s)}^2 \right] \tag{c}
\]

\[
\leq V_z^s(u) - V_{z_+}^s(u) + \frac{1}{2} \| \eta - \xi \|_{(s),*}^2 - \| w - z \|_{(s)}^2 \tag{d}
\]

Proof. By definition of \( z_+ = \text{Prox}_z(\eta) \) one can have \( \langle \eta - \omega_z^s(z) + \omega_z^s(z_+), u - z_+ \rangle \geq 0 \); rearranging terms and taking into account the definition of \( V_z^s(u) \) (7.37), one gets (a). By definition of \( w = \text{Prox}_z(\xi) \),

\[
\langle \xi - \omega_z^s(z) + \omega_z^s(w), z_+ - w \rangle \geq 0
\]

and

\[
\langle \eta, w - z_+ \rangle \leq \langle \eta - \xi, w - z_+ \rangle + \langle \omega_z^s(w) - \omega_z^s(z), z_+ - w \rangle
\]

Replacing the third term in the right-hand side of (a) with this upper bound and rearranging terms, one gets (b). (c) follows from (b) due to the strong convexity of \( \omega_z \) implying that \( V_z^s(u) \geq \frac{1}{2} \| u - z \|_{(s)}^2 \), and (d) is an immediate consequence of (c).

2°. Applying Lemma D.1 to \( z = z_{t,s}, \xi = \tau(t,s)G(z_{t,s}) \) (which results in \( w = w_{t,s} \)) and \( \eta = \tau(t,s)G(w_{t,s}) \) (which results in \( z_+ = z_{t+1,s} \)) and using (D.1):

\[
(a) \quad \tau(t,s)\langle G(w_{t,s}), w_{t,s} - u \rangle \leq V_{z_{t,s}}^s(u) - V_{z_{t+1,s}}^s(u) + \delta_{s} \forall u \in \mathcal{X},
\]

\[
(b) \quad \delta_{s} \leq \frac{1}{2} \left[ \tau(t,s)^2 \| G(w_{t,s}) - G(z_{t,s}) \|_{(s),*}^2 - \| w_{t,s} - z_{t,s} \|_{(s)}^2 \right] \tag{D.2}
\]

Observe that (D.2.b) combines with (7.35) and (7.36.b) to imply that \( \delta_{s} \leq 0 \) whenever \( \tau(t,s) = \tau(s) \), as claimed in Proposition 7.1.
Appendix D. Proofs for the Saddle Point Algorithm

30. Let \( u \in \mathcal{Z}_s \), and let
\[
\overline{\phi}_s(x) = \max_{v \in \mathcal{Y}} \phi(x, v).
\]
Summing up (D.2a) over \( t = 1, \ldots, N_s \), taking into account that \( V_{z,s}^f(u) \leq 1 \) due to \( u \in \mathcal{Z}_s \) by (7.36a), that \( V_{z}^f(u) \geq 0 \), and setting \( \lambda_t = \tau(t, s)/\sum_{t=1}^{N_s} \tau(t, s) \), one gets \( \lambda_t \geq 0, \sum_{t=1}^{N_s} \lambda_t = 1 \), and
\[
\forall u \in \mathcal{Z}_s : \sum_{t=1}^{N_s} \lambda_t \langle G(w_{l,t}), w_{l,t} - u \rangle \leq A := \frac{1}{\sum_{t=1}^{N_s} \tau(t, s)} \tag{D.3}
\]
(recall that \( \delta_t, s \leq 0 \) by the premise of Proposition (7.1)). On the other hand, setting \( w_{l,t} = (x_{l,t}, v_{l,t}), u = (x, v) \) and noting that \( z^o = (x^o, v^o) = \sum_{t=1}^{N_s} \lambda_t w_{l,t} \) (see (7.40)) and hence :
\[
\sum_{t=1}^{N_s} \lambda_t \langle G(w_{l,t}), w_{l,t} - u \rangle \\
= \sum_{t=1}^{N_s} \lambda_t \left[ \langle \phi'_i(x_{l,t}, v_{l,t}), x_{l,t} - x \rangle + \langle \phi'_j(x_{l,t}, v_{l,t}), v - v_{l,t} \rangle \right] \\
\geq \sum_{t=1}^{N_s} \lambda_t \left[ \phi(x_{l,t}, v_{l,t}) - \phi(x, v_{l,t}) \right] + \left[ \phi(x_{l,t}, v) - \phi(x_{l,t}, v_{l,t}) \right] \tag{a} \tag{D.4} \\
= \sum_{t=1}^{N_s} \lambda_t \left[ \phi(x_{l,t}, v) - \phi(x_{l,t}, v_{l,t}) \right] \\
\geq \phi(\sum_{t=1}^{N_s} \lambda_t x_{l,t}, v) - \phi(x, \sum_{t=1}^{N_s} \lambda_t v_{l,t}) = \phi(x^o, v) - \phi(x, v^o) \tag{b}
\]
(inequalities in (a), (b) are due to the convexity-concavity of \( \phi \), so that (D.4) results in \( \phi(x^o, v) - \phi(x, v^o) \leq A \) for all \( (x, v) \in \mathcal{Z}_s \). Taking supremum in \( (x, v) \in \mathcal{Z}_s \), one can be arrived at
\[
\overline{\phi}_s(x^o) - \phi(v^o) \leq A \leq \frac{L_s}{N_s}, \tag{D.5}
\]
where the concluding inequality follows from the definition of \( A \) due to \( \tau(t, s) \geq \tau(s) = 1/L_s \) for all \( t \).

40. Observe that the left-hand side in (D.5) is \( \geq \overline{\phi}_s(x^o) - \text{SadVal} \) (due to \( \phi(v^o) \leq \text{SadVal} \)), while the right-hand side in (D.5) is \( \leq \frac{\theta R_s^2}{32} \) due to (7.35), (7.38). Thus, (D.5) implies that
\[
\overline{\phi}_s(x^o) - \text{SadVal} \leq \frac{\theta R_s^2}{32}. \tag{D.6}
\]
Claim is that in fact $\bar{\phi}_s(x^s) = \bar{\phi}(x^s)$. Indeed, assuming the opposite, let $\mathcal{Y}_s = \{v \in \mathcal{Y} : \|v - \bar{v}_s\|_v \leq R_s\}$, and let $\bar{v} = \arg\max_{v \in \mathcal{Y}_s} \phi(x^s, v)$, so that $\bar{\phi}_s(x^s) = \phi(x^s, \bar{v})$. Since $\bar{\phi}(x^s) := \max_{v \in \mathcal{Y}} \phi(x^s, v) > \bar{\phi}_s(x^s) := \max_{v \in \mathcal{Y}_s} \phi(x^s, v)$ and $\mathcal{Y}_s$ is cut off $\mathcal{Y}$ by the inequality $\|v - \bar{v}_s\|_v \leq R_s$, $\|\bar{v} - \bar{v}_s\|_v = R_s$, while by (I_s) one gets $\|v_s - \bar{v}_s\|_v \leq R_s/2$, whence, in particular, $v_s \in \mathcal{Y}_s$ and $\|v_s - \bar{v}_s\|_v \geq R_s/2$.

Since the function $\phi(x^s, v)$ is strongly concave, modulus $\theta$ w.r.t. $\|\cdot\|_v$, and attains its maximum in $v \in \mathcal{Y}_s$ at $\bar{v}$, while $v_s \in \mathcal{Y}_s$, one gets $\phi(x^s, v_s) \leq \phi(x_s, \bar{v}) - \frac{\theta}{2} \|v_s - \bar{v}\|_v^2 \leq \phi(x^s, \bar{v}) - \frac{\theta}{8} R_s^2$. It follows that $\text{SadVal} = \min_{x \in \mathcal{X}} \phi(x, v_s) \leq \phi(x^s, \bar{v}) - \frac{\theta}{8} R_s^2 = \bar{\phi}_s(x^s) - \frac{\theta}{8} R_s^2$. The resulting inequality contradicts (D.6), and this contradiction shows that in fact $\bar{\phi}_s(x^s) = \bar{\phi}(x^s)$. Thus, (D.5) reads

$$\bar{\phi}(x^s) - \bar{\phi}(v^s) \leq \frac{L_s}{N_s} \leq \frac{\theta R_s^2}{32},$$

as required in (7.41) (recall that by construction $R_s = 2^{-s} R_0$). Finally, (D.7) implies that

$$\bar{\phi}(v_s) - \bar{\phi}(v^s) = \text{SadVal} - \bar{\phi}(v^s) \leq \bar{\phi}(x^s) - \bar{\phi}(v^s) \leq \frac{\theta R_s^2}{32},$$

since the function $\bar{\phi}(\cdot)$ is strongly concave, modulus $\theta$ w.r.t. $\|\cdot\|_v$, and attains its maximum over $v \in \mathcal{Y}$ at $v_s$, one gets $\bar{\phi}(v_s) - \bar{\phi}(v^s) \geq \frac{\theta}{2} \|v_s - v^s\|_v^2$, which combines with (D.8) to imply that $\|v_s - v^s\|_v \leq R_s/4 = R_{s+1}/2$; this is nothing but (I_{s+1}). Thus, it is proved that if (I_s) takes place, then Algorithm MPb ensures (7.41) and (I_{s+1}). Since (I_0) holds true by assumption, one can conclude that (7.41) and (I_s) take place for all $s$. All remaining claims in Proposition are straightforward.
Appendix E

Projection on the SVM constraint set

This appendix discusses the projection step encountered in Section 7.3.3. It considers the following problem where \( \mathcal{V} = \{ v \in \mathbb{R}^n | 0 \leq v_i \leq C, \sum_{i=1}^n v_i y_i = 0 \} \) is the SVM constraint set where \( y_i \) could be 1, or -1.

\[
\arg\min_{v \in \mathcal{V}} \left\{ -v^T q + \frac{1}{2} (v - \bar{v})^T (v - \bar{v}) \right\}, \tag{E.1}
\]

It is given that \( \bar{v} \in \mathcal{V} \)

\[
v_{+i} = \begin{cases} 
0, & \bar{v}_i + q_i + v y_i \leq 0 \\
C, & \bar{v}_i + q_i + v y_i \geq C \\
\bar{v}_i + q_i + v y_i, & \text{otherwise}
\end{cases} \tag{E.2}
\]

where \( \sum_i v_i y_i = 0 \) is required.

To this end, at first the feasible region of \( v \) is decided and then appropriate \( v^* \) is found by grid search.

\( i_+ \in \{ i \mid y_i = +1 \} \) In this case update formula is \( v_{+i} = \bar{v}_i + q_i + v \). Two extreme impossible cases are all \( v_{+i+} = 0 \) or \( v_{+i+} = C \)

- when \( v_{+i+} = 0 \ \forall i_+ \Rightarrow v \leq -\max_{i_+} (\bar{v}_i + q_i) \).
- when \( v_{+i+} = C \ \forall i_+ \Rightarrow v \geq C - \min_{i_+} (\bar{v}_i + q_i) \).
In this case update formula is \( v_{+i} = \bar{v}_i + q_i - v \). Two extreme impossible cases are all \( v_{+i} = 0 \) or \( v_{+i} = C \):

- when \( v_{+i} = 0 \) \( \forall i \) \( \Rightarrow \) \( v \geq \max_i (\bar{v}_i + q_i) \).
- when \( v_{+i} = C \) \( \forall i \) \( \Rightarrow \) \( v \leq \min_i (\bar{v}_i + q_i) - C \).

Hence lower range of \( v = \min(-\max_i (\bar{v}_i + q_i), \min_i (\bar{v}_i + q_i) - C) \) and upper range of \( v = \max(C - \min_i (\bar{v}_i + q_i), \max_i (\bar{v}_i + q_i)) \). Here a grid search can be done for searching closest \( v \) for which \( v_{+i}y_i = 0 \) holds.

If there are more than one solution satisfying the constraint \( v_{+i}y_i = 0 \) the solution which gives minimum objective value \( \frac{1}{2}(v - \bar{v})^T (v - \bar{v}) - q^T v \) should be taken.
Appendix F

Specific Kernels

For our comparison weighted Pairwise Distance Substructure Kernel described in [16] has been used. These kernels are purely based on protein structure alignment (specially aligned position of $c^\alpha$). For sake of completion the kernels is briefly described here, for more details please see [16]. Let $N_1$ and $N_2$ be the sub-structures, each having $l$ residues. Then kernels on protein structure are defined as:

F.1 Weighted Pairwise Distance Kernel

The kernel is defined as $K_{pds} = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} K_{pds}(N_{ia_i}, N_{ib_j}) \times K_{pds}(N_{ja_j}, N_{jd_i}) \times K_{norm}((N_{ia_i}, N_{ib_j}), (N_{ja_j}, N_{jd_i}))$ (F.1)

It is easy to see that $K$ is a positive semidefinite. A slight perturbation in position of any residue of one protein results in substantial changes in kernel values. It is thus useful to study the applicability of RSVM or USSVM procedure for such classifiers.
Bibliography


