Analysis and Extensions of Universum Learning

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Sauptik Dhar

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Advisor: Vladimir S. Cherkassky

January, 2014

Acknowledgements

First of all, I would like to express my deepest thanks and appreciation to my academic advisor, Prof. Vladimir Cherkassky. He has been my guide, my friend, and my inspiration throughout my PhD life. I sincerely appreciate his precious advice and consistent encouragement during my PhD study. It has been a pleasure and an honor working with him.

I would also like to thank my PhD committee members, Professor Xiaotong Shen, Professor Jarvis Haupt and Professor Nikolaos Sidiropoulos, for reviewing my thesis and giving valuable feedback and advice on my work.

I thank the Data Mining Research Team at Robert Bosch LLC, for providing a beautiful environment for research and collaboration during my internship. Specially, I would like to thank Dr. Caio Soares for his guidance and advice that helped me gain insights to solve real life data mining problems.

I thank my former and current lab mates: Dr. Feng Cai, Dr. Yunqian Ma, Wuyang Dai, Han-Tai Shiao, Tuo Zhao, Jieun Lee, Thomas Vacek, Adarsh Sivasankaran, and many others for their help and friendship.

Finally, I express my deepest gratitude to Ayan Paul, Puja Das, Sanjoy Dey, Subhrajit Roychowdhury, Sudhir Kudva and Sounak Basu for their wonderful friendship. And especially to Sohini Roy Chowdhury for her unwavering company. “This journey of thousand miles; felt like a single step” with such a companion around.

Dedication

To my parents and my brother Shouvik.

And to all my fellow PhD mates; who share a similar “Life of a PhD!”

```
function mylife()

%-------------------------------------------------------
% AUTHOR: Sauptik Dhar.
%-------------------------------------------------------

What else to say...

     Abt my life 2day!

Just been a CTRL C and CTRL V

     Of Yesterday!

Such a never ending loop

     A while(1);

With a single fprintf('Work ...

        ... until done!');

Not a single break; or a continue;

The only termination in case: 'You bid adieu';

How else do I change function mylife();

     And do {fun = INF;}, while(alive).
```

Abstract

Many applications of machine learning involve sparse high-dimensional data, where the number of input features is larger than (or comparable to) the number of data samples. Predictive modeling of such data sets is very ill-posed and prone to overfitting. Standard inductive learning methods may not be sufficient for sparse high-dimensional data, and this provides motivation for non-standard learning settings. This thesis investigates such a new learning methodology called Learning through Contradictions or Universum Learning proposed by Vapnik (1998, 2006) for binary classification. This method incorporates a priori knowledge about application data, in the form of additional Universum samples, into the learning process.

However, such a new methodology is still not well-understood and represents a challenge to end users. An overall goal of this thesis is to improve understanding of this new Universum learning methodology and to improve its usability for general users. Specific objectives of this thesis include:

- Development of practical conditions for the effectiveness of Universum Learning for binary classification.
- Extension of Universum Learning to real life classification settings with different misclassification costs and unbalanced data.
- Extension of Universum Learning to single-class learning problems.
- Extension of Universum Learning to regression problems.

The outcome of this research will result in better understanding and adoption of the Universum Learning methods for classification, single class learning and regression problems, common in many real life applications.

Table of Contents

List of Tables

List of Figures

Notation

The following notation is used throughout the thesis. Scalars are indicated by script letters such as a. Vectors are indicated by lowercase bold letters, such as $\mathbf{w}$. Matrices are given using uppercase bold letters $\mathbf{V}$. When elements of a matrix are accessed individually, we use the corresponding lowercase script letter. For example, the (i, j) element of the matrix $\mathbf{V}$ is v_{ij}.

n	Number of samples
d	Number of input variables, also dimensionality of the input/feature space
$\mathbf{x}$	Input column data vector
y	Output of a learning system. For classification y is a binary label, for regression y is real-valued.
$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$	Matrix of input samples
$\mathbf{y} = [y_1, \ldots, y_n]$	Vector of output samples
$f(\mathbf{x}, \omega)$	A class of approximating functions indexed by abstract parameter ω (ω can be a scalar, vector or matrix)
$f(\mathbf{x}, \omega^*)$	Model estimated from finite training data
Ω	Set of parameters, as in $\mathbf{w} \in \Omega$
$L(y, f(\mathbf{x}, \omega))$	Loss function
$(\mathbf{a} \cdot \mathbf{b})$	Inner (dot) product of two vectors

$R(\omega)$	Expected risk as a function of parameters
$R_{emp}(\omega)$	Empirical risk as a function of parameters
$K(\mathbf{x}_1, \mathbf{x}_2)$	General kernel function
Δ	Margin size parameter in delta–margin hyper plane
ε	parameter in epsilon-insensitive loss function
ξ	Error between the target function and the approximating function. Also used to denote additive noise for regression data.

In additional to the above notations, there are chapter-specific notations which will be introduced locally in each chapter.

Chapter 1 Introduction

There is a growing need for development of powerful and robust methods for estimating models from data. In many applications, good models are defined in terms of their generalization capability, where the goal is to estimate unknown dependency from available (training) data, in order to use this model for predicting future (test) samples. Most learning methods developed in statistical learning, pattern recognition and machine learning are based on standard inductive formulation of the learning problem (Vapnik 1998, Cherkassky and Mulier 2007, Hastie et al 2001). That is, a given application is usually formalized as either standard classification or regression problem. This standard inductive learning setting follows a mathematical framework for predictive learning (Vapnik 1998) shown in Fig. 1.1. Under standard inductive formulation, a learning machine observes finite number of training samples, aka training data $(\mathbf{x}_i, y_i), (i = 1,..., n)$, and the goal of learning is to estimate unknown mapping $f: \mathbf{x} \to y$ in order to imitate the system's response for future (test) inputs. This function is selected from a set of admissible functions $f(\mathbf{x}, \omega)$ implementable by the Learning Machine. For regression problems, the system's output is real-valued $y \in R$ and for (binary) classification problems, the output is a class label $y \in \{+1, -1\}$. Hence, the problem of learning can be stated as function estimation from finite training data. The quality of 'useful' models is defined via a loss function quantifying the discrepancy between model's prediction and the true output. So the goal of learning is defined as minimization of the prediction risk functional. The learning system shown in Fig. 1.1 suggests that the goal of learning is to

'imitate' the output of unknown system, in the sense of minimization of prediction error. This goal of *system imitation* is different from the goal of *system identification* (e.g., estimation of a probabilistic model) adopted in classical statistics (Vapnik 1998, 2006).

A solid theoretical framework for predictive learning based on the risk minimization approach is provided by Vapnik-Chervonenkis theory. VC-theory makes a strong argument that for finite sample estimation problems one should always use the most appropriate *direct formulation* of the learning problem. This principle can be also applied on the methodological level of formalizing application-domain requirements (Vapnik 2006; Cherkassky 2001, 2007, 2013). That is, for a given application, one should first choose/introduce an appropriate learning problem formulation (reflecting application domain requirements), and only then develop (or select) learning algorithms for this learning formulation.

Non-standard learning formulations can be also motivated by many real-life applications where the dimensionality of data samples d is (much) larger than the training sample size n. For such sparse settings (common in genomics, medical imaging, document classification etc.), application of classical statistical methods usually fails. Likewise, direct application of Support Vector Machines (SVM) usually fails, due to very ill-posed nature of estimation problems. That is, according to VC-theory (Vapnik 1998), there are three factors responsible for generalization of large-margin separating hyperplanes:

– small ratio m/n, e.g., small number of support vectors m relative to the sample size n,

– large margin size relative to the radius r of the sphere containing all training samples, and

– small dimensionality of the input space *d* (relative to sample size *n*), i.e., $d << n$.

Classical approaches rely on the third factor (small number of features); whereas the SVM approach relies on the first two factors. Of course, the 'best' strategy for generalization is often data-dependent. Due to geometric properties of very high-dimensional data ($d >> n$), most samples tend to become support vectors, and the radius r grows faster than margin (Cherkassky and Mulier, 2007). Hence, good generalization using linear SVMs in the input space may become difficult or impossible. Properties of sparse high-dimensional data lead to the effect known as *data piling* (Ahn and Marron, 2005), which helps to explain why many classification methods (regularized LDA, SVM, least squares SVM) provide similar generalization performance for high-dimensional data sets.

Most approaches to learning with high-dimensional data focus on improvements to *existing inductive methods* (i.e., LDA or SVM) that try to incorporate a priori knowledge about the good models (i.e., via specially designed kernels for SVM methods). These approaches, however, are fundamentally constrained by the inductive learning setting itself. In contrast, *non-inductive learning* methodologies focus on the most appropriate *direct formulation* of the learning problem. It can be argued that most recent advances in statistical learning (i.e., transduction, semi-supervised learning, co-clustering, multi-task learning) reflect improved understanding of the learning problem setting (Vapnik 2006, Cherkassky and Mulier 2007).

For example, consider the task of hand-written digit recognition. Under *standard inductive learning* setting one has to estimate the class decision boundaries, given labeled training samples. Then the quality (classification accuracy) of a classifier is measured using an independent test set. The non-standard learning setting investigated in this thesis assumes that along with labeled training data (i.e., handwritten digits) we have additional a priori knowledge (e.g. in the form of *handwritten letters*). These handwritten letters reflect the style of writing and thus can potentially improve generalization of a classifier for digit recognition. This approach leads to *Learning through Contradiction*, or the *Universum learning* (Vapnik 2006).

This thesis aims to extend the methodology of Universum learning, originally proposed for classification, to other types of learning problems. The rest of the thesis is organized as follows,

Chapter 2 presents Universum-SVM for binary classification which motivates the proposed research. We propose practical conditions for the effectiveness of U-SVM and provide empirical results in support of the practical conditions.

Chapter 3 extends Universum-SVM for cost-sensitive settings. Empirical results are provided in support of the practical conditions for the effectiveness of the cost-sensitive U-SVM.

Chapter 4 describes an extension of the Universum learning to single-class estimation problems and provides practical conditions for its effectiveness. Empirical results are provided in support of the practical conditions.

Chapter 5 describes new learning formulation called the Universum regression and the corresponding Universum-SVM regression formulation. Empirical comparisons between standard SVM regression and U-SVM regression are also presented.

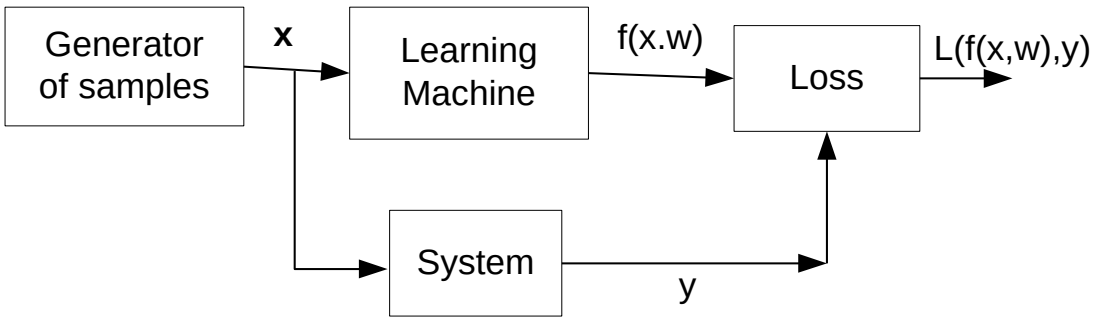


Figure 1.1: Generic system for inductive learning

Chapter 2 Universum Learning for Classification

2.1 Introduction

Sparse high-dimensional data is common in modern machine learning applications. In micro-array data analysis, technologies have been designed to measure the gene expression levels of tens of thousands of genes in a single experiment. However, the sample size in each data set is typically small ranging from tens to low hundreds due to the high cost of measurements. Similarly, in brain imaging studies the dimensionality of the input data vector is larger than the sample size. Such sparse high-dimensional problems represent new challenges for classification methods.

Most approaches to learning with high-dimensional data focus on improving existing inductive methods that try to incorporate a priori knowledge about the optimal model (Cherkassky and Mulier, 2007; Hastie et al, 2001; Schölkopf and Smola, 2002). Common examples include:

- clever preprocessing and feature extraction techniques that incorporate application-domain knowledge into the selection of a small number of informative features;
- selection of good kernels in SVM methods;
- specification of the prior distributions in Bayesian methods.

These techniques have been successfully used in many real-life applications (Camps-Valls et al, 2007).

Another approach to such ill-posed high-dimensional problems is to use non-standard learning settings that incorporate a priori knowledge about application data

and/or the goal of learning directly into the problem formulation. In order to illustrate several non-standard methodologies, consider the task of hand-written digit recognition (Cherkassky and Mulier, 2007). Under a standard inductive learning setting, one has to estimate the class decision boundaries from labeled examples of handwritten digits. Then the prediction accuracy of a classifier is measured using an independent test set. Under the transductive (Vapnik, 1998) and semi-supervised learning settings (Chapelle et al, 2006), the learning system uses both labeled (training) and unlabeled (test) samples, in order to predict class labels for future inputs. Under the setting called Learning with Structured Data (Vapnik, 2006), the training data originates from t different persons (groups), and this additional information (about group labels) is incorporated into learning. Here the goal of learning is to estimate a single predictive model, since the group labels are not provided for test inputs. Another possible scenario is to assume that both the training and test data are generated by t persons, and that the group label is known for both training and test data. This setting known as Multi-Task Learning (MTL) requires estimation of t related classifiers (Caruana, 1997; Evgeniou and Pontil, 2004; Liang and Cherkassky, 2008). Yet another modification of standard inductive learning assumes that along with labeled training data (i.e., handwritten digits) one has additional a priori information in the form of other handwritten letters. These handwritten letters reflect the style of handwriting and can potentially improve generalization. This leads to the setting known as Learning through Contradiction, or learning in the Universum environment (Vapnik, 2006). Such non-standard learning settings reflect properties of real-life applications, and can result in improved generalization, relative to standard

inductive learning. However, these new methodologies are more complex, and their advantages and limitations are not well understood.

The idea of 'inference through contradictions' was introduced by Vapnik (1998, 2006) in order to incorporate a priori knowledge into the learning process. Recall that standard inductive learning methods introduce a priori knowledge about the space of admissible models. It may be argued that in real applications (especially with sparse high-dimensional data) such 'good' parameterizations are hard to come by. However, it may be feasible to introduce a priori knowledge about admissible data samples. These additional unlabeled data samples (called virtual examples or the Universum) are used along with labeled training samples, to perform an inductive inference. Examples from the Universum are not real training samples. However, they reflect a priori knowledge about application domain. For example, if the goal of learning is to discriminate between handwritten digits 5 and 8, one can introduce additional 'knowledge' in the form of other handwritten digits 0, 1, 2, 3, 4, 6, 7, 9. These examples from the Universum contain certain information about handwritten digits, but they cannot be assigned to any of the two classes (5 or 8). Also note that Universum samples do not have the same distribution as labeled training samples.

Vapnik (2006) introduced Universum learning for an SVM based approach called Universum-SVM (U-SVM). Following this, there have been numerous work (Weston et al, 2006; Bai and Cherkassky, 2008) focused on the algorithmic implementation of the U-SVM, and its empirical validation. These studies confirmed that Universum learning can improve generalization performance, especially for sparse high-dimensional data.

However, the obtained performance strongly depends on a good choice of the Universum. More recent studies have proposed and analyzed criteria for a good choice of a Universum (Zhang et al, 2008; Chen and Zhang, 2009; Sinz et al, 2008). First, Sinz et al (2008) showed that the optimal decision boundary of the U-SVM tends to make the normal vector orthogonal to the principal direction of the Universum data set This condition holds for both the original Vapnik's Universum formulation (Vapnik, 2006) and for the least-squares U-SVM (Sinz et al, 2008), where the squared loss function is adopted for both labeled and Universum samples. Further, they show the connection (equivalency) between the least-squares U-SVM and the maximization of an explicit analytic criterion. Later, Chen and Zhang (2009), proposed a graph-theoretic index for measuring the 'in-betweenness' of Universum samples. However, they assume semi-supervised learning framework, and use squared loss in their SVM-style optimization formulation. Their approach aims at selecting a portion of the Universum data set that is 'useful' for boosting generalization performance.

In this chapter we pursue the same general objective as (Sinz et al, 2008), i.e. the characterization of a good Universum for Vapnik's original formulation. However, we take a more practical and specific approach. That is, we ask the following questions:

i. Can a given Universum data set improve generalization performance of standard SVM classifier trained using only labeled data?
ii. Can we provide practical conditions for (i), based on the geometric properties of the Universum data and labeled training data?

This approach is more suitable for non-expert users, because practitioners are interested in using U-SVM only if it provides an improvement over the standard SVM.

The chapter is organized as follows. Section 2.2 describes the standard SVM formulation. Section 2.3 explains Vapnik's (2006) original formulation for Universum-SVM. The practical conditions for the effectiveness of U-SVM are provided in Section 2.4. Next we provide empirical results to illustrate these conditions, using both synthetic and real-life data sets in Section 2.5. Section 2.6 provides analytic interpretation of these conditions, by relating them to analytic conditions in Sinz et al. (2008). Finally, conclusions are presented in Section 2.7.

2.2 SVM for Classification

In this chapter, we deal with binary classification. The output of a binary classification system takes on only one of two values $y = \{+1, -1\}$ corresponding to two classes. Therefore, in a learning machine, $D(\mathbf{x}, \omega), \omega \in \Omega$ are a set of indicator functions. A commonly used loss function for such binary classification problems is the misclassification error:

$$L(D(\mathbf{x}, \omega), y) = \begin{cases} 0 & \text{if } y = D(\mathbf{x}, \omega), \\ 1 & \text{if } y \neq D(\mathbf{x}, \omega). \end{cases} \tag{2.1}$$

Let's consider a binary classification setting where we are given finite training data $(\mathbf{x}_i, y_i), i = 1, \ldots, n,$ with $\mathbf{x} \in R^d$ and $y \in \{+1, -1\}$. The goal of SVM is to find the optimal decision function $D(\mathbf{x}, \omega) = sign(\mathbf{w} \bullet \mathbf{x} + b)$ with good generalization performance.

Assuming that training data is linearly separable, there are many separating hyperplanes ($f(\mathbf{x},\omega)=\mathbf{w}\bullet\mathbf{x}+b$) satisfying the constraints $y_i(\mathbf{w}\bullet\mathbf{x}_i+b)\geq 1,\ i=1,\ldots,n.$ SVM approach considers an optimal separating hyperplane (Vapnik, 1998), for which the margin (i.e. the distance between the closest data points to the hyperplane) is maximized. SVM implements structural risk minimization (SRM) inductive principle by keeping the value of empirical risk fixed (i.e. zero for linearly separable case) and minimizing the confidence interval (by maximizing margin). The concept of margin is illustrated in Figure 2.1. Maximization of margin is equivalent to minimization of $\|\mathbf{w}\|$. To this end, SVM solves the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to:} \quad & y_i(\mathbf{w}\bullet\mathbf{x}_i+b)\geq 1,\ i=1,\ldots,n. \end{aligned} \tag{2.2}$$

When training data is not linearly separable then training samples are allowed to fall inside the margin (so called soft margin). Non-negative slack variables $\xi_i=\max(1-y_i f(\mathbf{w},\mathbf{x}_i),0),\ i=1,\ldots,n,$ which represent deviations from the margin borders, are introduced (see Figure 2.2). Empirical risk is then defined as: $R_{\text{emp}}(\mathbf{w})=\sum_{i=1}^{n}\xi_i$. In this case, SVM attempts to strike a balance between the goal of minimization of empirical risk and maximizing the margin:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to:} \quad & y_i(\mathbf{w}\bullet\mathbf{x}_i+b)\geq 1-\xi_i,\ i=1,\ldots,n. \end{aligned} \tag{2.3}$$

In this form, the coefficient $C\geq 0$ controls the trade-off between complexity and proportion of non-separable samples and must be determined by model selection.

Problem (2.3) is a quadratic programming (QP) problem, which is typically solved in its dual form (according to convex optimization, the primal and dual forms of QP are equivalent):

$$\begin{aligned} \min_{\alpha} \quad & -\sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \quad 0 \le \alpha_i \le C, \; i = 1, \ldots, n. \end{aligned} \tag{2.4}$$

Formulation (2.4) has the solution in the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x} \bullet \mathbf{x}_i) + b \tag{2.5}$$

In the expansion (2.5), the sample points with non-zero α_i are called support vectors (SVs). The bias term b is given by

$$b = y_s - \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x}_i \bullet \mathbf{x}_s)$$

where $(\mathbf{x}_s, y_s)$ is one of the support vectors.

2.3 Universum-SVM for Binary Classification

The idea of Universum learning was introduced by Vapnik (1998, 2006) to incorporate a priori knowledge about admissible data samples. It was originally introduced for binary classification, where in addition to labeled training data we are also given a set of unlabeled examples from the Universum. The Universum contains data that belongs to the same application domain as the training data. However, these samples are known not to belong to either class. These Universum samples are incorporated into learning as

explained next. Let us assume that labeled training data is linearly separable using large margin. Then the Universum samples can fall either inside the margin or outside the margin borders (see Fig. 2.3). Under U-SVM we favor hyperplane models where the Universum samples lie inside the margin, as these samples do not belong to either class. Such Universum samples (inside the margin) are called contradictions, because they are falsified by the model (i.e., have non-zero slack variables for either class labels).

Next, we briefly review the optimization formulation for Universum SVM classifier (Vapnik, 2006; Weston. et al, 2006). Let us consider an inductive setting (for binary classification), where we have labeled training data $(\mathbf{x}_i, y_i)$, $i = 1, 2, ...n$ and a set of unlabeled examples $(\mathbf{x}_j^*)$, $j = 1, 2, ...m$ from the Universum. The Universum contains data that belongs to the same application domain as the training data, but these samples are known not to belong to either class. The optimization formulation for U-SVM (Vapnik, 2006; Weston et al , 2006), is presented in (2.6). This formulation is shown only for linear parameterization; however it can be generalized to nonlinear cases using kernels. Here, for labeled training data, we use standard SVM soft-margin loss with slack variables ξ_i. The Universum samples $(\mathbf{x}_j^*)$ are penalized using an ε–insensitive loss (shown in Fig. 2.4). Let ξ_j^* denote slack variables for samples from the Universum. Then the U-SVM formulation is given as:

$$\min_{\mathbf{w},b} \qquad R(\mathbf{w},b) = \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + C\sum_{i=1}^{n}\xi_i + C^*\sum_{j=1}^{m}\xi_j^* \tag{2.6}$$

subject to constraints: for labeled data: $y_i[(\mathbf{w}\cdot\mathbf{x}_i)+b] \geq 1-\xi_i \;\; \xi_i, \geq 0, i = 1,...,n$

for the Universum: $\left|(\mathbf{w}\cdot\mathbf{x}_j^*)+b\right| \leq \varepsilon + \xi_j^* \;\; \xi_j^* \geq 0,\ j = 1,...,m$

Here $\varepsilon \geq 0$ is user-defined and usually set to zero or some small fixed value. Parameters $C, C^* \geq 0$ control the trade-off between the margin size, the number of errors and the number of contradictions.

The solution to the optimization problem (2.6) defines the large margin hyperplane $f(\mathbf{x}) = (\mathbf{w}^* \cdot \mathbf{x}) + b^*$ that incorporates a priori knowledge (i.e., Universum samples) into the final model. The dual formulation for inductive SVM in the Universum environment, and its nonlinear (kernelized) version can be obtained using optimization theory and standard SVM techniques, where the decision function in the dual space is constructed by using a kernel matrix of both the labeled samples and the Universum samples (Weston et al, 2006). This quadratic optimization problem is convex due to convexity of the constraints for labeled data and for the Universum. Efficient computational algorithms for solving this problem involve modifications of standard SVM software (Weston et al, 2006). The U-SVM software is available at: http://www.cs.unibo.it/~roffilli/sw.html.

2.4 Practical Conditions for the Effectiveness of Universum-SVM

There are *two design factors* necessary for a successful practical application of U-SVM.

- *Model Selection*: which becomes rather difficult because the kernelized U-SVM has 4 tunable parameters: C, C^*, kernel parameter and ε (in contrast, standard SVM has only two tuning parameters).
- generalization performance of U-SVM may be also affected by a bad choice of the Universum data.

In practice, it may be difficult to separate these two factors. Hence we propose the following strategy for judging the effectiveness of a given Universum. This strategy is based on analysis of the histogram of projections of the training and universum samples onto the normal direction of the SVM decision boundary and is provided next:

STRATEGY TO ANALYZE THE EFFECTIVENESS OF U-SVM

2.4 a estimate SVM classifier for a given (labeled) training data set This step involves model selection of the C and kernel parameter.

2.4 b generate low-dimensional representation of training data by projecting it onto the normal direction vector of SVM hyperplane estimated in (a) (see Fig. 2.5).

2.4 c project the Universum data onto the normal direction vector of the SVM hyperplane (see Fig. 2.6 a).

2.4 d analyze the histogram of projected Universum data in relation to projected training data (see Fig. 2.6 b).

The benefits of this strategy are two-fold. First, it simplifies the characterization of good Universum data. Specifically, based on the statistical properties of the projected Universum data relative to labeled training data (in step 2.4 d), we can formulate the conditions on whether using this Universum will improve the prediction accuracy of standard SVM estimated in step 2.4 a. The practical conditions for the effectiveness of U-SVM are provided next and illustrated in Fig. 2.7.

PRACTICAL CONDITIONS FOR EFFECTIVENESS OF U-SVM	
A1)	The histogram of projections of training samples is separable, and its projections cluster outside the SVM margin borders denoted as points -1/+1 in the projection space.
The histogram of projections of the Universum data:	
A2)	is symmetric relative to the (standard) SVM decision boundary, and
A3)	It has wide distribution between the SVM margin borders.

The second aspect of the proposed strategy is simplified model selection. Specifically, this strategy involves two steps, i.e.

- perform model selection for the C and kernel parameters for the standard SVM classifier (in step 2.4 a).
- perform model selection for C*/C (ratio) while keeping C and kernel parameters fixed (as in (a)). Parameter ε is usually pre-set to a small value and does not require tuning.

This strategy is used in all empirical comparisons reported in Section 2.5 below (where parameter ε is set to zero).

2.5 Empirical Results for U-SVM

This section presents empirical results to illustrate the conditions (A1)-(A3) for the effectiveness of Universum SVM. The *first set of experiment* uses the synthetic 1000-dimensional hypercube data set, where each input is uniformly distributed in [0, 1]

interval and only 200 out of 1000 dimensions are relevant for classification. An output class label is generated as $y = \text{sign}(x_1+x_2+\ldots+x_{200} - 100)$. For this data set, only linear SVM is used because the optimal decision boundary is known to be linear. The training set size is 1,000, validation set size is 1,000, and test set size is 1,000. For U-SVM, 1,000 synthetic Universum samples are generated by using a commonly used strategy called Random Averaging (RA) (Weston et al, 2006). For RA, the Universum samples are generated by randomly selecting positive and negative training samples and computing their average (as illustrated in Fig. 2.8).

Next we provide the modeling results for standard SVM and U-SVM using linear kernel. The model selection is performed by tuning parameter values providing the smallest error on the independent validation set Table 2.1 shows performance comparison for the standard SVM, and U-SVM. The table shows the average value of the test error over 10 random experiments. Here, for each experiment we randomly select the training/validation set, but use the same test set The standard deviation of the test error is shown in parenthesis. The histogram of projections for linear SVM is provided in Fig. 2.9. An analysis of the histograms indicates that the training samples are not separable and hence U-SVM will not provide any improvement over SVM. This is consistent with empirical results shown in Table 2.1.

The second set of experiments involves classification of handwritten digits '5' and '8' using the MNIST data (http://www.cs.nyu.edu/~roweis/data.html). Here each sample is represented as a real-valued vector of size 28*28=784. On average, approximately 22% of the input features are non-zero which makes this data very sparse. The goal is to

investigate the effectiveness of three types of Universa: handwritten digits 1, 3 and 6, and to explain their effectiveness by analyzing histograms of projections of both labeled and Universum data sets. For this experiment:

- Training/validation set samples size is 1000 (500 per class);
- Universum set sample size is 1,000 (digits '1', '3', '6' and RA). Note, we keep the number of Universum samples equal to the number of training samples. Increasing the number of Universum samples does not change the histogram of projections significantly. Hence, the relative performance of the different types of the Universum remains the same.
- Test set sample size is 1,866.

Previous studies suggest that RBF kernel $K(\mathbf{x},\mathbf{x}') = \exp\left(-\gamma\left\|\mathbf{x}-\mathbf{x}'\right\|^2\right)$ works well for this dataset (Schölkopf and Smola, 2002). Model selection for standard RBF SVM classifier and for U-SVM is performed using the independent validation data set Each experiment is repeated 10 times with different random realizations of training/validation/Universum samples, and the average test error (and its standard deviation) is reported. The test error rates of SVM and U-SVM are shown in Table 2.2, and the typical histograms of projections for training data and Universum data are shown in Fig. 2.10.

Typical histograms of projections shown in Fig. 2.10 suggest that digit 1 Universum is less effective than digit 3 or 6 because it has more biased distribution between projections of labeled data, i.e., digits 5 and 8. The Universum samples for digits 3 and 6 are more widely and symmetrically distributed inside the margin borders, so they are

expected to provide better performance (than digit 1 Universum). These findings are consistent with the empirical results in Table 2.2, showing no statistically meaningful improvement for digit 1 Universum, and a good improvement for digits 3 and 6.

For the third set of experiments we use the Real-life ABCDETC data set, where data samples represent the handwritten lower case letters 'a' and 'b'. Each sample is represented as a real-valued vector of size 100*100 = 10000. The task involves classification of handwritten characters 'a' and 'b' (Weston et al, 2006). The goal is to investigate the effectiveness of three types of Universa: 'All upper case letters from A to Z', 'All digits from 0 to 9' and 'Random Averaging' of training data.

For this experiment:

- Training/validation set samples size is 150 (75 per class);
- Universum set sample size is 1,500;
- Test set sample size is 209, i.e., 105 samples from class 'a' and 104 from class 'b'.

For this data set, we use a 3rd degree Polynomial Kernel following (Weston et al, 2006). Model selection for the standard Polynomial SVM classifier and for U-SVM is performed using the validation data set Each experiment is repeated 10 times with different random realizations of training/validation/Universum samples, and the average test error (and its standard deviation) is reported. The test error rates of SVM and U-SVM are shown in Table 2.3, and typical histograms of projections for training and Universum data are shown in Fig. 2.11.

The histograms in Fig. 2.11 show that for both the 'Upper case letters A-Z' and 'digits 0-9' the Universum samples have a wider distribution than the Universum samples

obtained via Random Averaging. Hence, we can expect both 'Upper case letters A-Z' and 'digits 0-9' to be more effective than RA. This is confirmed by the empirical results in Table 2.3.

In summary, these results suggest that the Universum distribution should be wide enough, relative to the margin borders of standard SVM model estimated from labeled training data. The 'good' Universum helps to stabilize SVM decision boundary, and makes it less sensitive to random variability of training samples.

2.6 Analytic Interpretation

This section establishes the connection between our practical conditions for the effectiveness of Universum learning and recent analytic results. Sinz et al (2008) analyzed the geometric relations of the decision hyperplane learnt with the U-SVM to the Universum data set, and showed that the optimal solutions tend to make the normal vector orthogonal to the principal directions of the Universum. That is, under optimization formulation (1), the U-SVM algorithm 'tries to find a direction $\mathbf{w}^*$ such that the variance of the projections of the Universum samples on that direction is small' (Sinz et al, 2008). As argued earlier in Section 1, this insight is not very practical, because it does not explicitly describe the properties of Universa *in relation to the labeled training data*. In fact, according to the U-SVM formulation (2.6) an optimal direction $\mathbf{w}^*$ tries to achieve two goals:

1. Separate labeled samples with large margin (as in standard SVM);
2. Minimize the variance of Universum samples.

Under high-dimensional settings, labeled training data tends to be separable (in some optimally chosen kernel space), so the first goal can be achieved by a standard SVM. This motivates a two-step strategy (shown in Section 2.4), where the standard SVM is estimated first, and then the conditions for the effectiveness of a Universum (i.e., for goal 2) are stated (in A1-A3)**.** This incremental strategy also alleviates the problem of model selection, because parameters of standard SVM are tuned separately.

Further, our conditions (A1)-(A3) for the effectiveness of a Universum implement the above cited analytic property that the optimal direction vector $\mathbf{w}^*$ minimizes the variance of the projections of Universum samples (Sinz. et al, 2008). Namely, our conditions apply to projections of Universum samples onto the vector $\mathbf{w}$ of the standard SVM model. Condition (A2) ensures that the mean of projected Universum samples falls close to SVM decision boundary, or equivalently that the mean of Universum is (approximately) the same as the mean of training samples. This is clearly necessary for minimizing the variance of projections of Universum according to (Sinz. et al, 2008). Condition (A3) ensures that Universum data can indeed provide an improvement relative to standard SVM. That is, if the Universum is narrowly distributed near SVM decision boundary, then the solution vector $\mathbf{w}$ of standard SVM would provide small variance of projections of the Universum, so that no additional improvement (due to this Universum) can be expected.

For the least-squares U-SVM, closed-form analytic interpretation becomes possible. Sinz et al (2008) showed an equivalency between the (least-squares) U-SVM learning

and the maximization of a hybrid Rayleigh's coefficient due to the kernel oriented Principal Component Analysis (kPCA) and kernel Fisher discriminant analysis (kFDA):

$$\max_{\mathbf{w}} \frac{\overbrace{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{b}}\mathbf{w}}^{\text{from kFDA}}}{\underbrace{C(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{w}}\mathbf{w})}_{\text{from kFDA}} + \underbrace{C^{*}\sum_{j=1}^{m}\mathbf{w}^{\mathrm{T}}(\mathbf{x}_{j}^{*}-\boldsymbol{\mu})(\mathbf{x}_{j}^{*}-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{w}}_{\text{from kPCA}}} \tag{2.7}$$

where,

$\mathbf{w} \equiv$ The normal weight vector of decision hyper plane.

$\boldsymbol{\mu} \equiv$The empirical mean of the training samples $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$.

$\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \equiv$ The empirical class means given by, $\boldsymbol{\mu}_c = \frac{1}{n_c}\sum_{i \in c}\mathbf{x}_i$ where c=Class -1,+1.

$\mathbf{S}_{\mathrm{b}} \equiv$ The between class scatter matrix; $\mathbf{S}_{\mathrm{b}} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^{\mathrm{T}}$.

$\mathbf{S}_{\mathrm{w}} \equiv$ The within class scatter matrix given by, $\mathbf{S}_{\mathrm{w}} = \sum_{c=-1,+1}\sum_{i \in c}(\mathbf{x}_i - \boldsymbol{\mu}_c)(\mathbf{x}_i - \boldsymbol{\mu}_c)^{\mathrm{T}}$.

$\mathbf{x}_j^{*} \equiv$ the universum samples, where $j = 1...m$.

$C, C^{*} \geq 0$, control for the tradeoff between minimization of errors and maximization of the number of contradictions.

Our conditions (A1)-(A3) can be only approximately related to the analytic criterion (2.7), because we use original U-SVM formulation (with hinge loss). Under our approach, the effectiveness of the Universum is evaluated relative to standard SVM

model estimated from labeled data (shown in Section 2.4). This approach can be interpreted using the analytic formulation (2.7) as follows:

- Minimize the term marked 'from kPCA', since the other two terms in (2.7) correspond to the solution provided by standard SVM, and they are fixed.

Then the universum samples contribute to the maximization of the hybrid Rayleigh's coefficient through the minimization of the term $\sum_{j=1}^{m} \mathbf{w}^{\mathrm{T}}(\mathbf{x}_j^* - \boldsymbol{\mu})(\mathbf{x}_j^* - \boldsymbol{\mu})^{\mathrm{T}}\mathbf{w}$. Further, this term can be rewritten as the sum of two terms:

$$\sum_{j=1}^{m} \mathbf{w}^{\mathrm{T}}(\mathbf{x}_j^* - \boldsymbol{\mu})(\mathbf{x}_j^* - \boldsymbol{\mu})^{\mathrm{T}}\mathbf{w} \tag{2.8}$$

$$= \mathbf{w}^{\mathrm{T}}[m(\boldsymbol{\mu}_{\mathrm{U}} - \boldsymbol{\mu})(\boldsymbol{\mu}_{\mathrm{U}} - \boldsymbol{\mu})^{\mathrm{T}}]\mathbf{w} + \sum_{j=1}^{m}[\mathbf{w}^{\mathrm{T}}(\mathbf{x}_j^* - \boldsymbol{\mu}_{\mathrm{U}})(\mathbf{x}_j^* - \boldsymbol{\mu}_{\mathrm{U}})^{\mathrm{T}}\mathbf{w}]$$

where, $\boldsymbol{\mu}_{\mathrm{U}} = \frac{1}{m}\sum_{j=1}^{m} \mathbf{x}_j^*$ is the mean of universum samples.

The first term in (2.8) is the squared distance (D^2) between the means of the Universum samples and training samples projected onto the normal weight vector ($\mathbf{w}$) of the standard SVM model. For high-dimensional data, most training samples cluster at/near the margins. So, for balanced data sets, the mean of the training samples is likely to be the standard SVM decision boundary. Hence, our condition (A2) is equivalent to the first term in (2.8), i.e. minimization of the projected distance (D) between the mean of the universum samples $\boldsymbol{\mu}_{\mathrm{U}}$ and the mean of the training samples $\boldsymbol{\mu}$. Because in the first term of (2.8), the distance between the means is very small, due to our condition (A2), maximization of the Rayleigh's coefficient (2.7) depends mainly on minimization of the second term, i.e. the variance of the universum samples projected onto the normal weight

vector. Thus, for a case where we have a wide distribution (larger variance) of the universum samples projected onto the normal weight vector, as stated in our condition (A3); we may expect to maximize the Rayleigh's coefficient in (2.7) by minimizing this large variance. On the other hand, if this variance is small, we expect no or little improvement from the Universum.

2.7 Conclusion

This chapter investigates the effectiveness of the U-SVM for finite-sample data. In general, performance of learning methods is always affected by the properties of application data at hand. New learning settings, such as U-SVM, are inherently more complex than standard SVM and they have more tuning parameters. So it is important to have practical criteria that ensure potential advantages of using U-SVM for a given data set This is a difficult problem, because the effectiveness of U-SVM depends on the properties of labeled data as well as Universum samples. Meaningful analytic characterization of such data sets is quite difficult. So we propose a novel representation of training data using projections of this data onto the normal direction of SVM decision boundary. Analysis of the univariate histograms of projections, presented in this paper, leads to practical conditions for the effectiveness of Universum learning. That is, a Universum data set is effective, if its univariate histogram of projections is symmetric and widely distributed, relative to (standard) SVM decision boundary.

Empirical results using several real-life and synthetic data sets illustrate the usefulness of the proposed approach, for several types of Universa, and several real-life

and synthetic data sets. Proposed practical conditions are also shown to be closely related to analytic conditions independently derived in (Sinz. et al). However, our conditions are more useful for practitioners than analytic criteria in (Sinz. et al), because our approach:

- Provides an explicit characterization of the properties of the Universum and the properties of labeled training data. These properties are conveniently represented in the form of univariate histograms;
- Directly relates prediction performance of U-SVM to that of standard SVM (using only labeled data);

Further, the proposed approach significantly simplifies model selection for U-SVM. That is, the regularization parameter C and the kernel parameter for the U-SVM formulation (2.6) are selected via training a standard SVM classifier (using only the labeled training data). Then model selection for U-SVM involves tuning only two remaining parameters, C*/C and ε.

In conclusion, we point out that most studies of the U-SVM use balanced data sets with equal misclassification costs. That is, the number of positive and negative labeled samples is (approximately) the same, and the relative cost of false positive and false negative errors is assumed to be the same. This chapter also assumes such a balanced setting, where false positive and false negative errors are assigned equal cost in the optimization formulation (2.6). Many practical applications involve unbalanced data and unequal costs. The next chapter presents Universum learning for classification under such cost-sensitive settings.

Table 2.1 Comparison of standard SVM and U-SVM on Synthetic Hypercube data using linear kernel

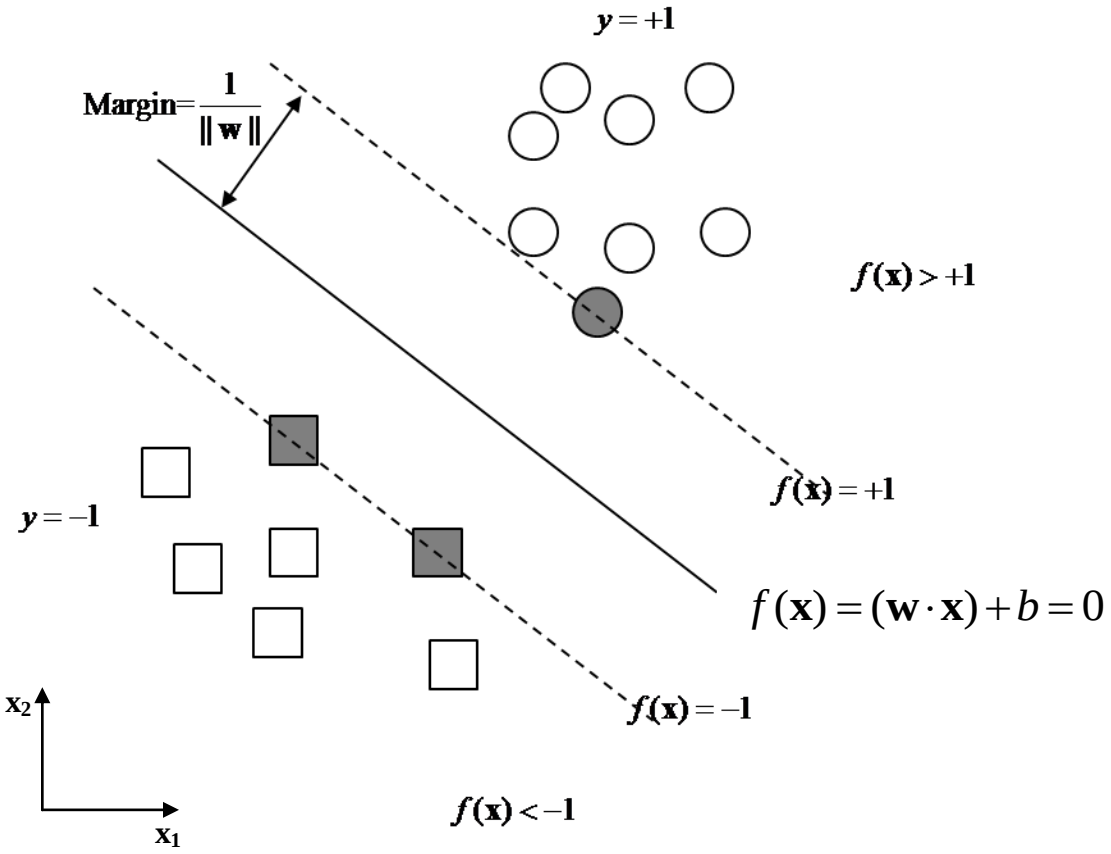
	SVM	U-SVM(RA)
Synthetic data (Linear)	26.63% (1.54%)	26.89% (1.55%)

Table 2.2 Test error rates for MNIST data with different Universa. Training set size is 1,000 samples.

	SVM	U-SVM (digit 1)	U-SVM (digit 3)	U-SVM (digit 6)
Test error	1.47% (0.32%)	1.31% (0.31%)	1.01% (0.28%)	1.12% (0.27%)

Table 2.3 Test error rates for ABCDETC data with different Universa. Training set size is 150 samples

	SVM	U-SVM (upper case)	U-SVM (all digits)	U-SVM (RA)
Test error	20.47% (2.60%)	18.42% (2.97%)	18.37% (3.47%)	18.85% (2.81%)



$$f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b = 0$$

Figure 2.1: Binary classification problem, where ‘ ◯ denotes samples from positive class and ‘ □ ’ denotes samples from negative class. The margin is the distance between the closest data points to the hyperplane.

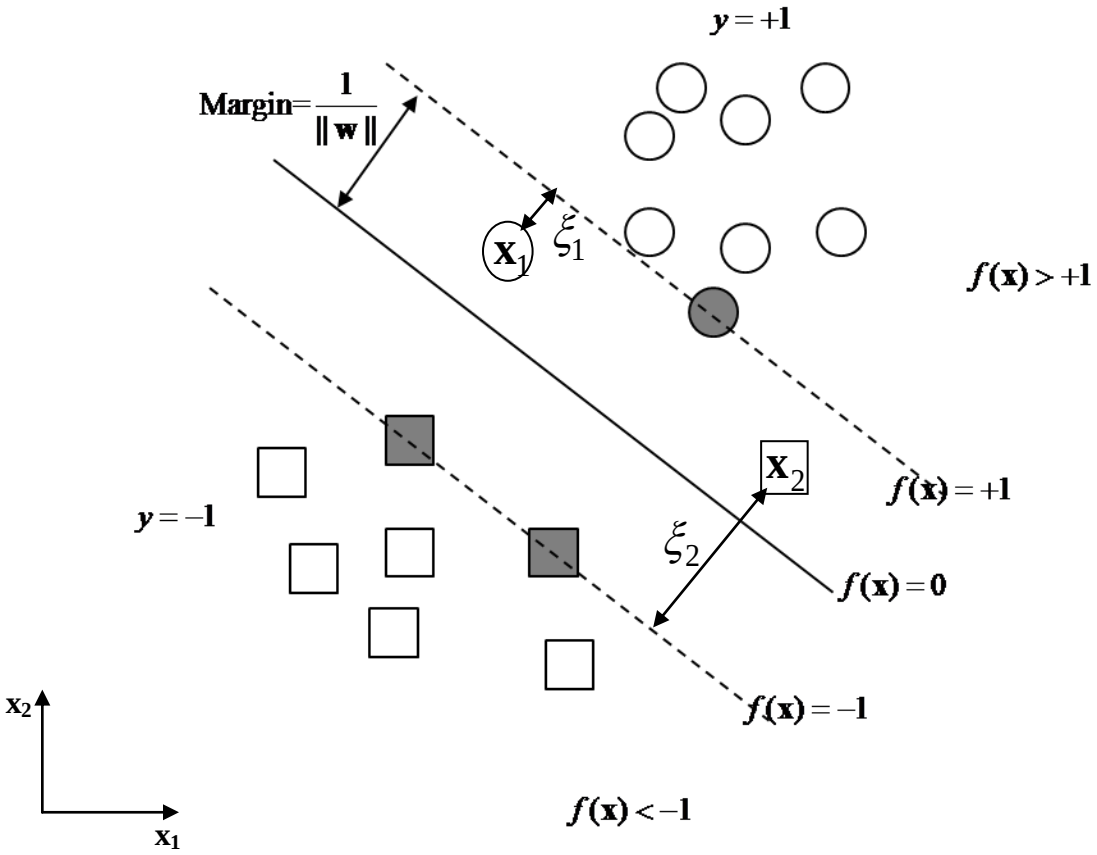


Figure 2.2: Non-separable case for binary classification. Slack variables $\xi_i = 1 - y_i f(\mathbf{x}_i)$ correspond to the deviation from the margin borders.

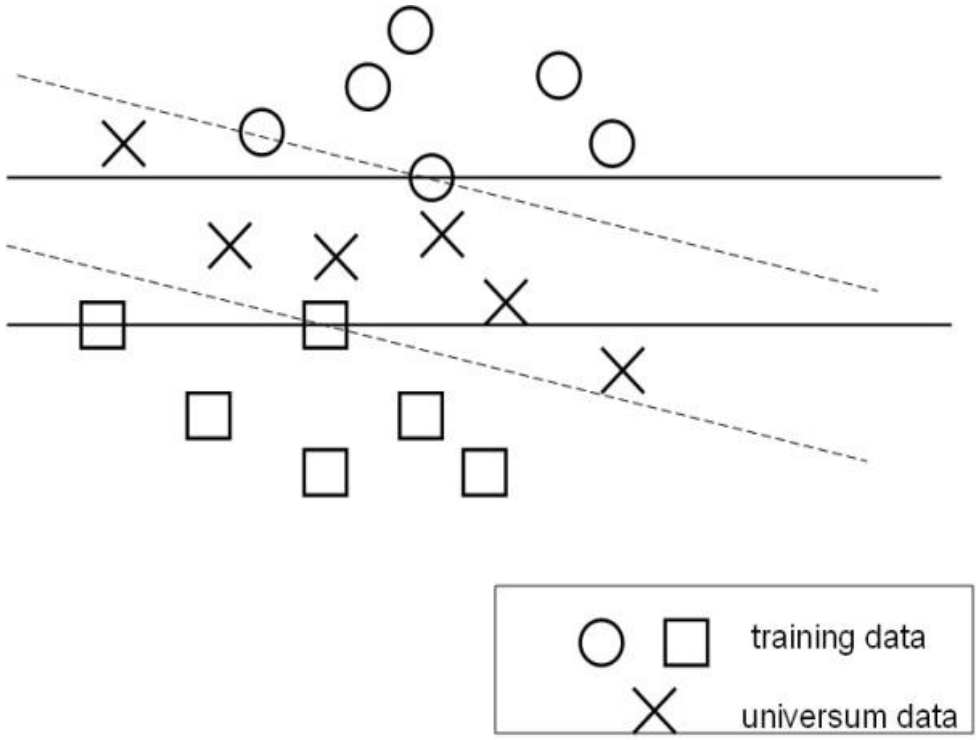


Figure 2.3: Two large-margin separating hyperplanes explain training data equally well, but have different number of contradictions on the Universum. The model with a larger number of contradictions should be favored.

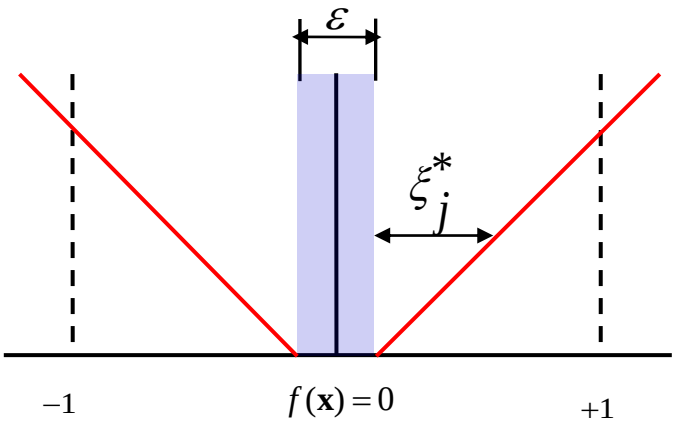
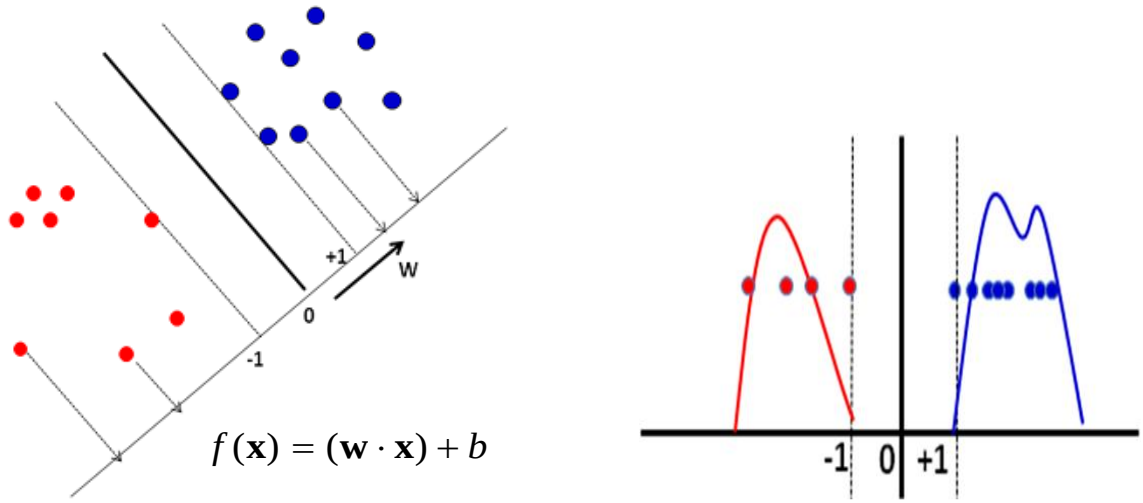
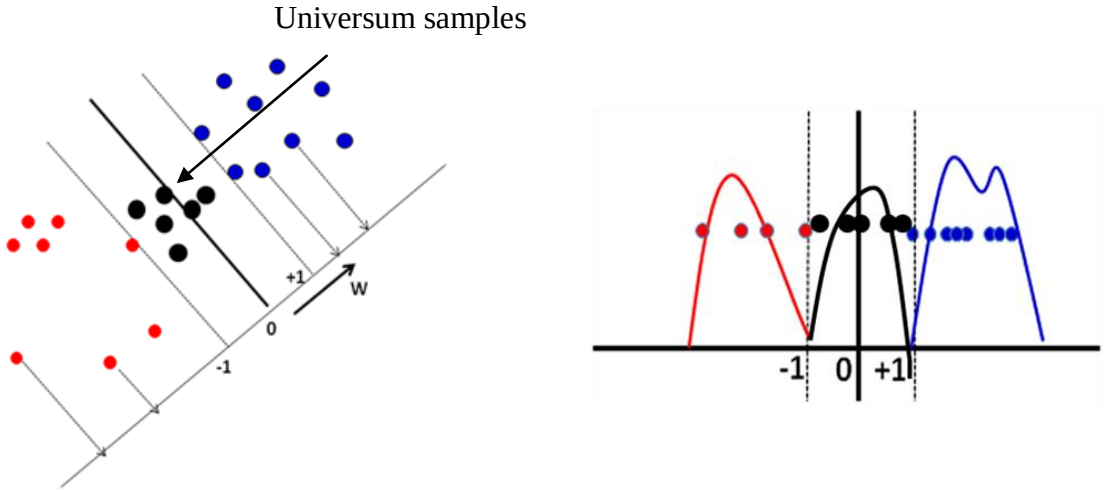


Figure 2.4: The ε - insensitive loss for the Universum samples. Universum samples outside the ε -insensitive zone are linearly penalized using the slack variables ξ_j^*.



(a) **(b)**

Figure 2.5: (a) Projection of the training data shown in red and blue onto the normal weight vector (w) of the SVM hyperplane. (b) Univariate histogram of projections. i.e. histogram of $f(\mathbf{x})$ values for training samples.



(a) **(b)**

Figure 2.6: (a) Projection of the universum data (shown in black) onto the normal weight vector (w) of the SVM hyperplane. (b) Univariate histogram of projections of the universum samples (shown in black) along with the training samples (shown in red/blue).

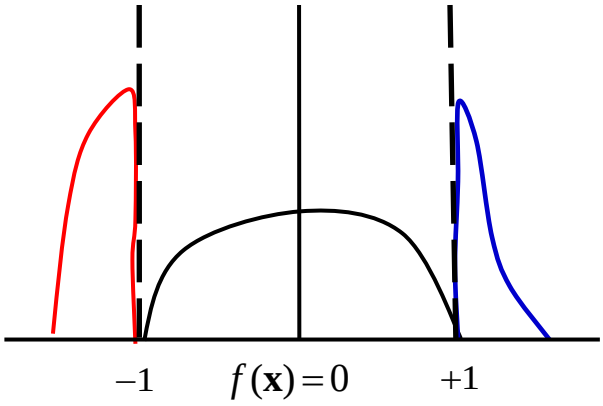


Figure 2.7: A schematic illustration of the histogram of projections of training and universum samples onto normal w vector of SVM decision boundary satisfying the practical conditions for the effectiveness of U-SVM.

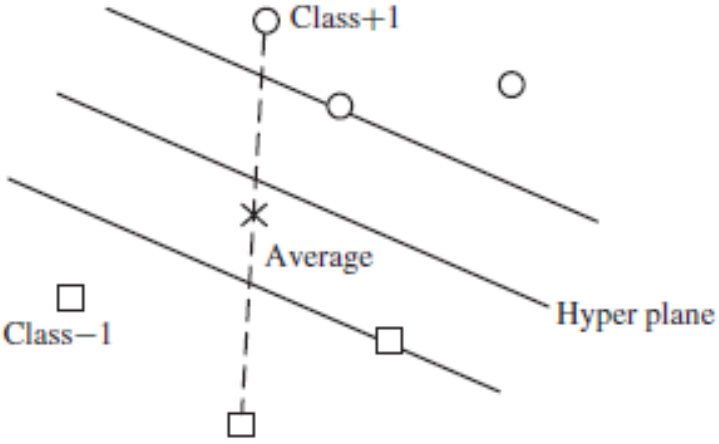


Figure 2.8: Generation of the Universum data by averaging.

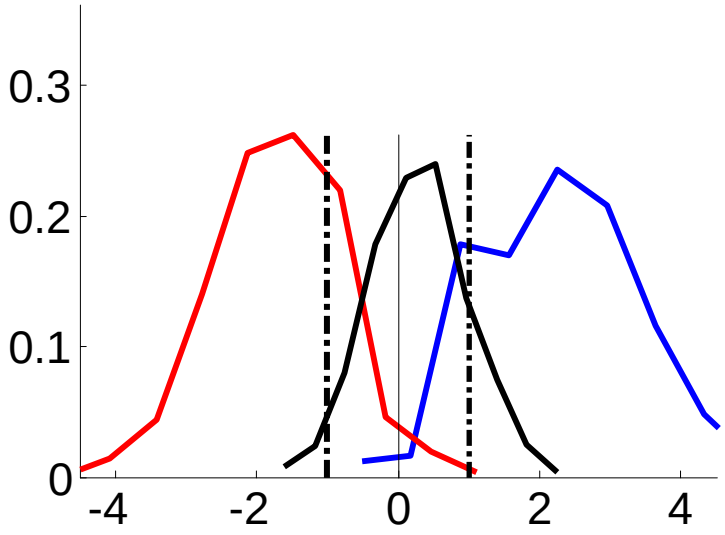
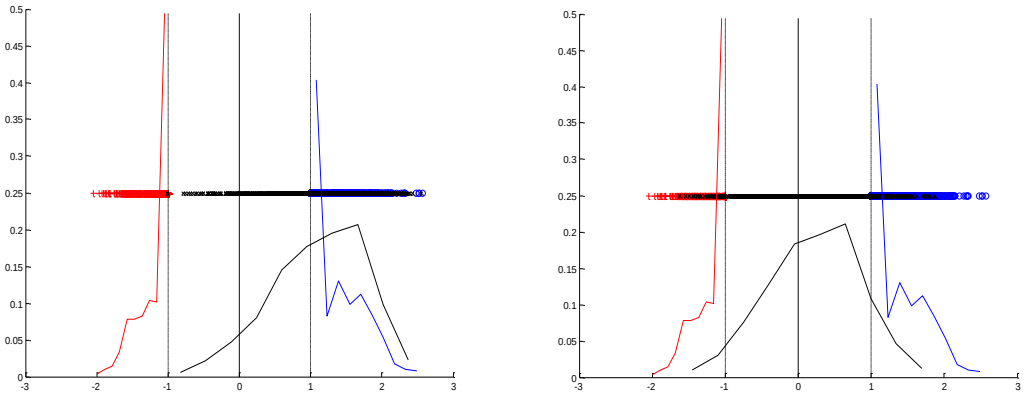
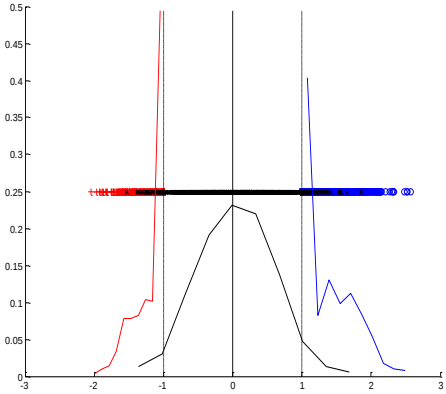


Figure 2.9: Histogram of projections onto normal direction of linear SVM for synthetic hypercube data set ($C=2^{-6}$ and $C^*/C=2^{-5}$).

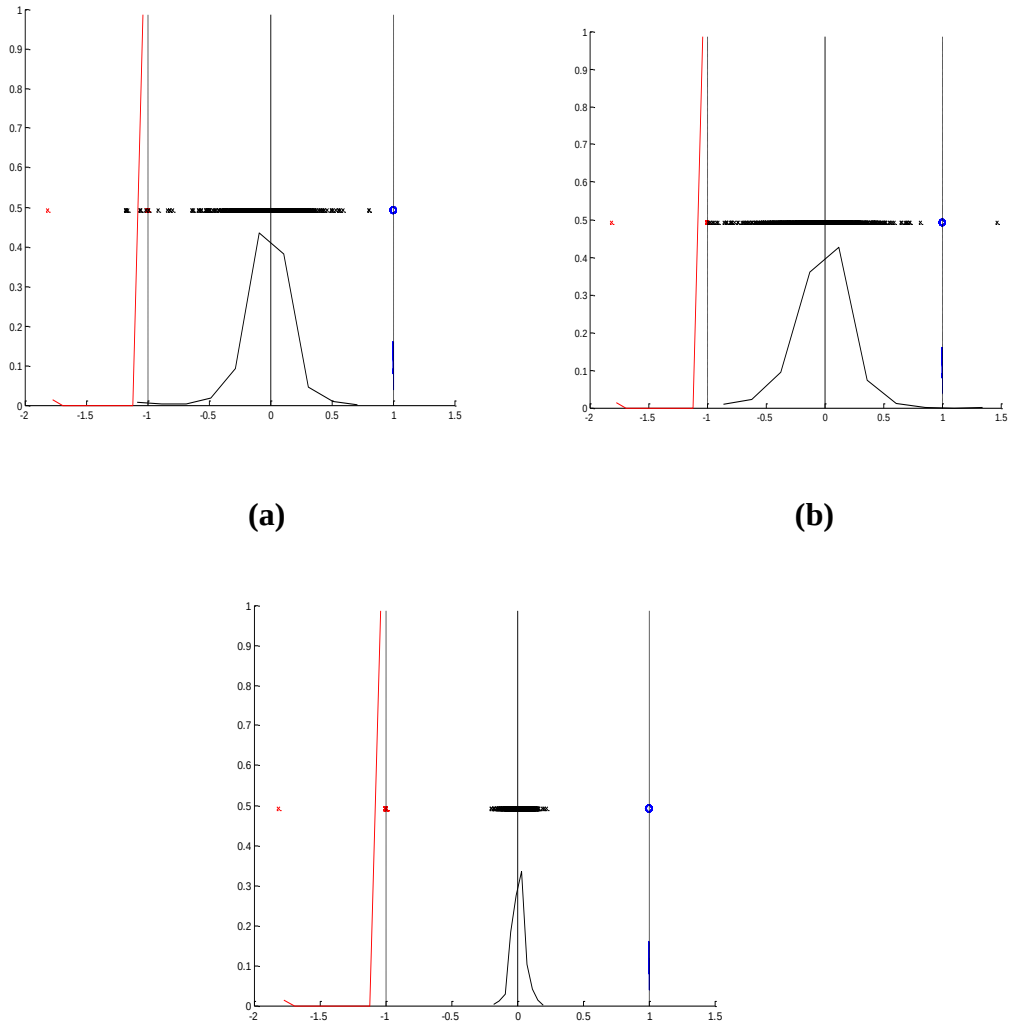


(a) **(b)**



(c)

Figure 2.10: Univariate histogram of projections onto (RBF) SVM normal weight vector (C=10, $\gamma = 2^{-6}$) for 3 different types of Universa for MNIST data. Training set size ~ 1,000 samples. Universum set size ~ 1,000 samples. (a) digit 1 Universum (C*/C=0.01) (b) digit 3 Universum (C*/C=0.1.) (c) digit 6 Universum (C*/C=0.01).



(a) **(b)**

(c)

Figure 2.11: Univariate histogram of projections onto (poly) SVM normal weight vector (C=10, d=3) for 3 different types of Universa for ABCDETC data. Training set size ~ 150 samples. Universum set size ~ 1,500 samples. (a) 'Upper case letters A to Z' Universum. (C*/C=0.1) (b) 'digits 0-9' Universum. (C*/C=0.1.) (c) RA Universum (C*/C=0.01).

Chapter 3 Cost-Sensitive Universum Learning for Classification

3.1 Introduction

As presented in Chapter 2, Universum learning is particularly effective for high-dimensional data settings, where the number of input features is comparable to the number of data samples used for model estimation. However, all previous studies on Universum learning use balanced data sets with equal misclassification costs (Weston et al, 2006; Cherkassky et al, 2011; Cherkassky and Dai, 2009; Sinz et al, 2008; Gao et al,2009; Zhang et al, 2008; Chen and Zhang, 2009; Bai and Cherkassky, 2008; Shen et al, 2011). That is, the number of positive and negative labeled samples is (approximately) the same, and the relative cost of false positive and false negative errors is assumed to be the same. However, many practical applications involve unbalanced data and different misclassification costs, i.e., credit card fraud detection, intrusion detection, oil-spill detection, disease diagnosis etc (Tan et al, 2006; Weiss et al, 2007; Elkan, 2001). In order to incorporate apriori knowledge (in the form of Universum data), we need to extend the Universum learning to handle such cost-sensitive settings.

Researchers have introduced many techniques to deal with problems involving unequal misclassification costs and unbalanced data settings (Tan et al, 2006; Weiss et al, 2007; Elkan, 2001). Typically, these methods follow two basic approaches:

- *Cost-Sensitive Learning*, where the costs of misclassification and the ratio of imbalance in the data are introduced directly into the learning formulation.
- *Oversampling/ Undersampling*, where the training samples of a particular class are replicated to reflect different misclassification costs (Elkan, 2001).

In this chapter we follow the direct approach of introducing the cost-ratios into U-SVM formulation. Specifically, we describe the U-SVM classification setting, where different misclassification costs for *false-positive* vs. *false-negative* errors are given as the ratio $r = C_{fp} / C_{fn}$. We modify Vapnik's original formulation for U-SVM (Vapnik, 1998; 2006) to include different misclassification costs. Further, we provide characterization of a good Universum for the proposed cost-sensitive U-SVM. Our approach follows the same practical strategy as before and tries to address the following questions:

i. Can a given Universum data set improve generalization performance of the cost-sensitive SVM classifier (Cherkassky and Mulier, 2007; Lin et al, 2002) trained using only labeled data?
ii. Can we provide *practical conditions* for (i), based on the geometric properties of the Universum data and labeled training data?

This approach is more suitable for non-expert users, because practitioners are interested in using cost-sensitive U-SVM only if it provides an improvement over the standard cost-sensitive SVM. Our conditions for the effectiveness of cost-sensitive U-SVM extend the conditions for the effectiveness of the standard U-SVM introduced in Chapter 2.

This chapter is organized as follows. Section 3.2 describes cost-sensitive SVM formulation. Section 3.3 explains the new formulation for cost-sensitive Universum-SVM. The practical conditions for the effectiveness of cost-sensitive U-SVM are provided in Section 3.4. Next we provide empirical results to illustrate these conditions, using both synthetic and real-life data sets in Section 3.5. Finally, conclusions are presented in Section 3.6.

3.2 Cost-Sensitive SVM

Consider a binary classification problem where we have labeled training samples as in standard SVM described in Section 2.2. However, in cost-sensitive settings we assign different importance (or cost) to false positive and false negative errors, as specified by the ratio $r = C_{fp} / C_{fn}$. The goal in such cost-sensitive learning is to estimate a classifier that minimizes the weighted error for future test samples (Tan et al, 2006; Elkan, 2001; Cherkassky and Mulier, 2007).

$$\text{weighted test error } = C_{fp} P_{fp} + C_{fn} P_{fn} \tag{3.1}$$

Here P_{fp} and P_{fn} denote the probability (error rate) of false positive and false negative errors.

Standard SVM formulation can be adapted for cost-sensitive settings by introducing the cost-ratios directly into the SVM formulation (Cherkassky and Mulier, 2007; Lin et al, 2002). The cost-sensitive SVM formulation is provided next,

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=+class}\xi_i + C\sum_{i\in-class} r\xi_i \tag{3.2}$$

subject to: $y_i[(\mathbf{w}\cdot\mathbf{x}_i)+b] \geq 1-\xi_i \qquad \xi_i \geq 0, \qquad i=1,...,n \qquad$ where, $r = \frac{C_{fp}}{C_{fn}}$.

The proposed cost-sensitive SVM uses unequal costs for the two classes in the labeled training data. The samples of the negative class lying inside the soft-margin are penalized r times more than those of the positive class.

As in standard SVM, problem (3.2) is typically solved in its dual form. The dual functional remains same as in (2.4). The only modification required is to the constraints, so that it incorporates the cost-ratio $r = \frac{C_{fp}}{C_{fn}}$. The cost-sensitive SVM dual formulation is provided below,

$$\begin{aligned} \min_{\alpha} \quad & -\sum_{i=1}^{n}\alpha_i + \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0, \\ & 0 \leq \alpha_i \leq C, \quad i \in +class \\ & 0 \leq \alpha_i \leq rC, \; i \in -class \end{aligned} \tag{3.3}$$

Formulation (3.3) has the solution of the form

$$f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i y_i (\mathbf{x}\bullet\mathbf{x}_i) + b \tag{3.4}$$

In the expansion (3.4), the sample points with non-zero α_i are called support vectors (SVs). The bias term b is given by

$$b = y_s - \sum_{i=1}^{n} \alpha_i y_i (\mathbf{x}_i \bullet \mathbf{x}_s)$$

where $(\mathbf{x}_s, y_s)$ is one of the support vectors.

3.3 Cost-Sensitive U-SVM

Next we show how to extend Universum learning for cost-sensitive settings. As discussed before in Section 3.1, there are many techniques to deal with cost-sensitive settings (Tan et al, 2006; Weiss et al, 2007; Elkan, 2001). In this thesis we follow the direct approach of introducing the cost-ratio $r = C_{fp} / C_{fn}$ into the U-SVM formulation (2.6). This leads to the proposed modified cost-sensitive U-SVM formulation (3.5),

Given, labeled training samples $(\mathbf{x}_i, y_i)$, $i = 1, 2, ...n$ and unlabeled universum samples $(\mathbf{x}_j^*)$, $j = 1, 2, ...m$.

$$\min_{\mathbf{w},b} R(\mathbf{w}, b) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C \sum_{i \in +class} \xi_i + C \sum_{i \in -class} r\xi_i + C^* \sum_{j=1}^{m} \xi_j^* \qquad (3.5)$$

subject to constraints:

(*training samples*): $y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 - \xi_i$

(*universum samples*): $\left|(\mathbf{w} \cdot \mathbf{x}_j^*) + b\right| \leq \varepsilon + \xi_j^*$

$\xi_i \geq 0, i = 1, ..., n$; $\xi_j^* \geq 0, j = 1, ..., m$

Here parameters r and $\varepsilon \geq 0$ are user-defined. In all empirical results presented later in section 3.5, the value of ε is set to zero. Tunable regularization parameters $C, C^* \geq 0$ control the trade-off between minimization of cost-weighted errors, margin size and the maximization of the number of contradictions.

The proposed cost-sensitive U-SVM uses unequal costs for the two classes in the labeled training data, following (Cherkassky and Mulier, 2007; Lin et al, 2002). The samples of the negative class lying inside the soft-margin are penalized r times more than those of the positive class. However, the loss for the Universum samples remains the same as in the original formulation (2.6).

Following (Weston et al, 2006) this quadratic optimization problem (3.5) can be solved by introducing the Univerum samples twice with opposite labels and hence solving a modified cost-sensitive SVM problem,

Here we define, $\mathbf{x}_{n+j} = \mathbf{x}_j^*$ and $y_{n+j} = +1$, $j = 1, 2, ...m$

$\mathbf{x}_{n+j} = \mathbf{x}_j^*$ and $y_{n+j} = -1$, $j = m+1, m+2, ...2m$

Then (3.5) is equivalent to solving the following optimization problem,

$$\min_{\mathbf{w},b} R(\mathbf{w},b) = \frac{1}{2}(\mathbf{w}\cdot\mathbf{w}) + \hat{C}\sum_{i=1}^{n+2m} k_i\xi_i \qquad (3.6)$$

subject to constraints: $y_i[(\mathbf{w}\cdot\mathbf{x}_i)+b] \geq \rho_i - \xi_i$ with $\xi_i \geq 0,\ i = 1,...,n+2m$

here, $\rho_i = 1$ and $\hat{C} = C$; for $i = 1,...,n$

$\rho_i = -\varepsilon$ and $\hat{C} = C^*$; for $i = n+1,...,n+2m$

and, $k_i = \begin{cases} C_{fp}/C_{fn} & \text{if } y_i = -1 \ (i = 1,... n) \\ 1 & \text{otherwise} \end{cases}$

This problem (3.6) can be easily solved in the dual form by using the original U-SVM software (http://www.cs.unibo.it/~roffilli/sw.html) where the $\hat{C}$ penalty term for the negative samples is weighted by the factor $r = C_{fp}/C_{fn}$. Hence, the computational cost for solving the cost-sensitive U-SVM problem remains the same as for the standard U-SVM; which is in turn equivalent to solving the standard SVM problem with *n+2m*

samples (Weston et al, 2006). The modified cost-sensitive U-SVM software is made publicly available (http://www.ece.umn.edu/users/cherkass/predictive_learning/SOFTWARES.html). The solution to the optimization problem (3.6) defines the large margin hyper-plane $f(x) = (\mathbf{w}^* \cdot \mathbf{x}) + b^*$ that incorporates a priori knowledge (i.e., Universum samples) and also reflects different misclassification costs.

3.4 Practical Conditions for the Effectiveness of Cost-Sensitive U-SVM

As evident from (3.5), the cost-sensitive U-SVM has the same design issues as the original U-SVM, i.e., model selection and selection of good Universum. Hence, we adopt the same strategy used originally for standard U-SVM (see 2.4 (a)-(d)). However, now the univariate histogram is generated by projecting the training and universum samples onto the normal direction vector of the cost-sensitive SVM hyperplane.

STRATEGY TO ANALYZE THE EFFECTIVENESS OF COST-SENSITIVE U-SVM	
3.4 a	estimate cost-sensitive SVM classifier for a given (labeled) training data set This step involves model selection of the C and kernel parameter.
3.4 b	generate low-dimensional representation of training data by projecting it onto normal direction vector of cost-sensitive SVM hyperplane estimated in (3.4 a).
3.4 c	project the Universum data onto the normal direction vector of the cost-sensitive SVM hyperplane.
3.4 d	analyze the histogram of projected Universum data in relation to projected training data.

Based on this histogram of projections, the practical conditions for the effectiveness of cost-sensitive U-SVM are provided next (and illustrated in Fig. 3.1).

PRACTICAL CONDITIONS FOR EFFECTIVENESS OF COST-SENSITIVE U-SVM	
B1.	The histogram of projections of the training data is well separable, and the samples from the class with smaller misclassification cost, (i.e. '+'ve class when $r<1$) cluster outside the '+1' soft-margin.
Conditions for the histogram of projections of the Universum data:	
B2.	is *slightly* biased towards the class for which the misclassification cost is higher, (i.e. '–' ve class when $r<1$), and
B3.	is well spread *within* the class means of the training samples.

These new conditions (B1)-(B3) take into account the inherent 'bias' in the estimated predictive models under cost-sensitive settings, discussed in (Cherkassky and Dhar, 2010; Cai et al, 2010). Conditions (A1)-(A3) represent a special case of conditions (B1)-(B3) when the costs are equal ($r=1$). Further, as before this strategy leads to the following two-step model selection (parameter tuning) for the cost-sensitive U-SVM:

1 perform model selection for C and kernel parameters for the cost-sensitive SVM formulation. (These parameters are then fixed and used for the cost-sensitive U-SVM).

2 perform model selection for the C*/C parameter specific to the cost-sensitive U-SVM formulation, while keeping C and kernel parameters fixed. Parameter ε is usually pre-set to a small value and does not require tuning.

This strategy is used in all empirical comparisons reported in Section 3.5 below (where parameter ε is set to zero).

3.5 Empirical Results for Cost-Sensitive U-SVM

This section presents empirical results to illustrate the conditions (B1)-(B3) for the effectiveness of cost-sensitive Universum SVM. For our empirical comparisons, the misclassification costs are specified as the ratio $r = C_{fp} / C_{fn}$, and the weighted test error (3.1) is normalized by its maximum possible value ($C_{fp} + C_{fn}$), as shown next,

$$\textit{Normalized}\,(C_{fp}P_{fp} + C_{fn}P_{fn}) = \frac{r \times (n_{fp}/n^-) + (n_{fn}/n^+)}{r\,(n^-/n^-) + (n^+/n^+)} \qquad (3.7)$$

$$= \frac{r \times (n_{fp}/n^-) + (n_{fn}/n^+)}{r+1}$$

Here n_{fp}, n_{fn} denotes the number of false positive and false negative samples, and n^+, n^- denotes the number of positive and negative test samples. Such normalization limits the value of the weighted error to the range of [0, 1] which is the same range used in standard binary classification problems (with equal costs). In the rest of the chapter, we refer to this *normalized weighted error* as simply the *test error*.

Further, several alternative metrics have been used in literature to measure the performance of a classification model under unbalanced and unequal misclassification costs settings (Tan et al, 2006; Cherkassky and Mulier, 2007). This thesis advocates using cost-sensitive U-SVM only if it provides an improvement over the cost-sensitive SVM (Cherkassky and Mulier, 2007, Lin et al, 2002). Lin et al (2002) have shown that the minimizer of the expected value of the cost-sensitive SVM loss function follows the Bayes rule. This provides theoretical justification for using the empirical estimate of the Bayes Risk (i.e. weighted test error) for our empirical comparisons.

The first set of experiment uses the synthetic 1000-dimensional hypercube data set, where each input is uniformly distributed in [0, 1] interval and only 200 out of 1000 dimensions are relevant for classification. An output class label is generated as $y = \text{sign}(x_1+x_2+\ldots+x_{200} - 100)$. For this data set, only linear SVM is used because the optimal decision boundary is known to be linear. The training set size is 1,000, validation set size is 1,000, and test set size is 1,000. For U-SVM, 1,000 Universum samples are generated by using the commonly used strategy called Random Averaging (RA).

For this data set, we consider three different cost ratios r = 0.5, 0.2, 0.1 to capture the effect of varying cost settings. We model this data for the standard SVM, cost-sensitive SVM and cost sensitive U-SVM using linear kernel. The model selection is performed by tuning parameter values providing the smallest weighted error on an independent validation set

Table 3.1 shows performance comparison for the standard SVM, cost-sensitive SVM and the cost-sensitive U-SVM with different cost-ratios (r = 0.5, 0.2, 0.1). The table

shows the average value of the (normalized weighted) test error over 10 random experiments. Here, for each experiment we randomly select the training/validation set, but use the same test set. The standard deviation of the test error is shown in parenthesis. Additionally we provide the average False Positive and False Negative rates (in %) over these 10 random experiments. The typical histograms of projections for training data along with the Universum data are shown in Fig. 3.2. In all figures the training samples for the two classes are shown in red and blue with their respective class means shown by the dotted red/blue line. The projection of the universum samples are shown in black. Further, we also show the average of the two class means of the training samples in green. This helps to understand a projection bias of the universum samples towards positive or negative class. The typical histograms of projections (in Fig. 3.2) show that the training samples are not separable. Hence, based on our conditions we expect no improvement over the cost-sensitive SVM. This is confirmed by the results in Table 3.1. For this data set (with unequal costs) application of standard Universum-SVM does not improve generalization (relative to cost-sensitive SVM).

The second set of experiments uses the real-life handwritten digits "8" vs. "5" MNIST data (http://www.cs.nyu.edu/~roweis/data.html). The goal is accurate classification of digits "8" vs. "5", where each sample is represented as a real-valued vector of size 28x28=784. For this experiment, we use four types of Universa: handwritten digits "1", "3", "6" and RA and analyze their effectiveness using the histograms of projections of both labeled and Universum data sets. For this experiment,

– Number of training samples = 1000. (500 per class)

– Number of validation samples = 1000. (500 per class. This independent validation set is used for model selection)

– Number of test samples=1866.

– Number of Universum samples = 1000.

– Linear SVM parameterization is used.

We label the digit '8' as class '+1' and the digit '5' as class '–1' and use the following cost-ratio,

$$\frac{\text{missclassification cost for(truth=digit 5,prediction=digit 8)}}{\text{missclassification cost for(truth=digit 8,prediction=digit 5)}} = \frac{C_{fp}}{C_{fn}} = r$$

Performance comparisons between the standard SVM, cost-sensitive SVM and the cost-sensitive U-SVM for different types of Universa: digit "1", "3", "6" and RA with different cost-ratios (r=0.5, 0.2, 0.1) are shown in Table 3.2. The typical histograms of projections for training data along with the Universum data are shown in Figs. 3.3 and 3.4. For this data set the histograms of projections for the cost-ratio r= 0.2 are not shown, because they look very similar to those for r = 0.1. Analysis of the histograms indicates that the training samples are not separable and hence cost-sensitive U-SVM will not provide any improvement over the cost-sensitive SVM. This is consistent with empirical results shown in Table 3.2.

The *3rd set of experiments* uses the same *real-life* handwritten digits "8" vs. "5". However, here we use an RBF kernel of the form $K(\mathbf{x},\mathbf{x}') = \exp\left(-\gamma\|\mathbf{x}-\mathbf{x}'\|^2\right)$. Performance comparisons between standard SVM, cost-sensitive SVM and the cost-sensitive U-SVM for the different types of Universa: digit "1", "3", "6" and RA with

different cost-ratios (r = 0.5, 0.2, 0.1) are shown in Table 3.3. The typical histograms of projections for training data along with the Universum data are shown in Figs. 3.5 and 3.6. Note that we do not show the histograms for the cost-ratio r = 0.2, because they look similar to histograms for r = 0.1.

The histograms of projections in Figs 3.5-3.6 have the following characteristics,

- the training samples are well-separable.
- *digit* '1': *well spread* universum samples *outside* training samples' class means and *highly biased* towards '+' ve class.
- *digit* '3': *well spread* universum samples about training samples' class means and *slightly biased* towards '–' ve class.
- *digit* '6': *well spread* universum samples about training samples class means but *slightly biased* towards '+' ve class.
- *Random Averaging*: *well spread* universum samples about training samples' class means but *slightly biased* towards '+'ve class.

Practical conditions (B1)-(B3) indicate that for the given well-separable training samples (digit '8' vs. '5'); digit '3' is the best choice for Universum samples. Although, the digit '6' and RA universum are well-spread about the training samples' class means; yet they are slightly biased towards the '+'ve class. Further, the digit '1' is the worst choice as they are not well-spread about the training samples' class means, and are highly biased towards the '+' ve class. These findings are consistent with empirical results in Table 3.3, showing no statistically meaningful improvement for digit 1 Universum, and a good improvement for digits '3', digit '6' and RA.

The *4th set of experiments* uses the *Real-life ISOLET data set* (Fanty and Cole,1991), where the data samples represent speech signals of 150 subjects for the letters 'B' vs. 'V'. Here, each sample is represented by 617 features that include spectral coefficients, contour features, sonorant features, pre-sonorant features, and post-sonorant features (Fanty and Cole, 1991). We label the voice signals for 'B' as class '+1' and 'V' as class '−1'. The cost-ratio is specified as,

$$r = \frac{C_{fp}}{C_{fn}} = \frac{\text{missclassification cost(truth='V',prediction='B')}}{\text{missclassification cost(truth='B',prediction='V)}}$$

For this experiment we use,

- Number of Training samples= 100. (50 per class)
- Number of Universum samples = 300 (2 types of Universa: letters D, P and RA).
- Number of Test samples=500. (This independent test set is used for model selection)

Our initial experiments suggest that linear SVM works well for this dataset Comparisons of the (linear) standard SVM, cost-sensitive SVM and the cost-sensitive U-SVM for the different types of Universa: letters D, P and RA with different cost-ratios (r=0.5, 0.2, 0.1) are shown in Table 3.4. The typical histograms of projections for training data along with the Universum data for the cost-ratios (r=0.5, 0.1) are shown in Figs 3.7 and 3.8. For this dataset typical histograms of projections for the cost-ratio r=0.2 are very similar to r=0.1, and have been omitted. From these figures, it is clear that the training samples are well-separable. Analysis of projections for different types of universum samples shows that:

– letter ‘P’ has well spread projections between the training samples’ class means and are slightly biased towards the ‘-’ ve class.
– letter ‘D’ has narrower projections than the letter ‘P’ and are slightly biased towards the ‘+’ ve class.
– Random Averaging has narrower projections than the letter ‘P’ and are slightly biased towards the ‘+’ ve class.

Hence, based on conditions (B1)-(B3), letter ‘P’ is expected to be more effective than letter ‘D’ and RA. This is consistent with the empirical results in Table 3.4.

For our final set of experiments we use the real-life German Traffic Sign Recognition Benchmark (GTSRB) dataset (http://benchmark.ini.rub.de/?section=gtsrbandsubsection=dataset#resultanalysis). The task is to perform traffic sign classification between the images of the signs "50" vs. "80". These sample images are represented by their pyramid histogram of oriented gradients (PHOG) features (Shen et al, 2011, Bosch et al, 2007). We label the traffic sign '50' as class '+1' and the traffic sign ‘80’as class ‘-1'. The cost-ratio is specified as,

$$r = \frac{C_{fp}}{C_{fn}} = \frac{\text{missclassification cost(truth='80',prediction='50')}}{\text{missclassification cost(truth='50',prediction='80')}}$$

For this experiment:

– Number of Training samples= 200. (100 per class)
– Number of Validation samples = 200. (100 per class)
– Number of Universum samples =1000 (3 types of Universa: signs ‘30’, ‘60’, RA).

- Number of Test samples=2000.
- Dimension of each sample =1568. (PHOG features)

Initial experiments suggest that linear parameterization is optimal for this dataset; hence only linear kernel has been used in all comparisons. Performance comparisons between standard SVM, cost-sensitive SVM and cost-sensitive U-SVM for the different types of Universa: signs '30', '60' and RA with different cost-ratios (r = 0.5, 0.2, 0.1) are shown in Table 3.5. The typical histograms of projections for training data along with the Universum data are also shown in Fig. 3.9, 3.10 and 3.11. Analysis of projections for different types of universum samples shows that:

- sign '30' has well spread projections between the training samples' class means and slightly biased towards the '-' ve class.
- sign '60' has well spread projections between the training samples' class means and slightly biased towards the '-' ve class.
- Random Averaging has narrower projections than the signs "30" and "60", except for the cost-ratio r=0.1, for which it has well-spread projections about the training samples' class means.

Hence, for the cost-ratios r=0.5, 0.2 we can expect signs "30" and "60" to be more effective than RA. Further, for r=0.1 all the three types of Universum are likely to provide good generalization. This is consistent with the empirical results in Table 3.5.

3.6 Conclusion

Previous studies (Weston et al, 2006; Cherkassky et al, 2011; Cherkassky and Dai, 2009; Sinz et al, 2008; Gao et al, 2009; Zhang et al, 2008; Chen and Zhang, 2009; Bai and Cherkassky, 2008; Shen et al 2011) have demonstrated the effectiveness of the Universum learning for improving the generalization of SVM classifiers. However, all of these studies use balanced data sets with equal misclassification costs. This paper presented a new U-SVM formulation that incorporates different misclassification costs. The proposed cost-sensitive U-SVM can be implemented using minor modifications to existing U-SVM software. This modified software is made publicly available at : http://www.ece.umn.edu/users/cherkass/predictive_learning/SOFTWARES.html.

We presented practical conditions for the effectiveness of the cost-sensitive U-SVM using histogram of projections. These proposed conditions also hold for unbalanced data sets typically seen in many biomedical/bioinformatics applications. These conditions can be adopted by general users, because:

1 They provide an explicit characterization of the properties of the Universum relative to the properties of labeled training data. These properties are conveniently represented in the form of the univariate histogram of projections;

2 They directly relate prediction performance of cost-sensitive U-SVM to that of cost-sensitive SVM.

It is important to note that, according to our analyses, meaningful characterization of 'good' Universum is possible only in the context of a particular labeled training dataset

Finally, we point out that many applications involve extreme scenarios with very high cost ratios or extreme unbalance in the data (viz., anomaly detection). Such problems follow a different learning framework called single-class learning (Tan et al, 2006; Cherkassky and Mulier, 2007), that is discussed in the next chapter.

Table 3.1 Comparison of standard/cost-sensitive SVM and cost-sensitive U-SVM for synthetic data

METHODS	standard SVM	cost-sensitive SVM	cost-sensitive U-SVM (RA)
Cost-Ratio r=0.5			
test error (in %)	**27.81(1.86)**	**24.84(1.38)**	**25.15(1.14)**
FP rate (in %)	27.49(8.57)	42.26(6.03)	39.9(5.72)
FN rate (in %)	27.96(6.39)	16.07(4.27)	17.69(3.74)
Cost-Ratio r=0.2			
test error (in %)	**21.21(5.68)**	**15.09(0.67)**	**14.92(0.57)**
FP rate (in %)	61.01(37.66)	73.75(14.07)	72.23(12.03)
FN rate (in %)	13.34(14.09)	3.37(2.26)	3.47(2.18)
Cost-Ratio r=0.1			
test error (in %)	**15.48(8.68)**	**8.80(0.43)**	**8.93(0.74)**
FP rate (in %)	68.79(37.22)	96.25(9.83)	90.99(11.53)
FN rate (in %)	10.24(13.17)	0.27(0.8)	0.93(1.52)

Table 3.2 Comparison of standard SVM, cost-sensitive SVM and cost-sensitive U-SVM for real life MNIST data (using linear kernel).

METHODS	standard SVM	cost-sensitive SVM	cost-sensitive U-SVM (digit 1)	cost-sensitive U-SVM (digit 3)	cost-sensitive U-SVM (digit 6)	cost-sensitive U-SVM (RA)
	Cost-Ratio (r=0.5)					
test error (%)	**4.80(0.51)**	**4.40(0.38)**	**4.39(0.31)**	**4.36(0.32)**	**4.33(0.44)**	**4.37(0.46)**
FP rate (in %)	3.94(0.50)	5.67(1.48)	5.64(1.35)	6.00(1.37)	5.84(1.40)	5.54(1.23)
FN rate (in %)	5.29(0.81)	3.82(0.69)	3.82(0.67)	3.60(0.67)	3.63(0.75)	3.84(0.67)
	Cost-Ratio (r=0.2)					
test error (%)	**4.91(0.48)**	**3.15(0.22)**	**3.12(0.24)**	**3.13(0.17)**	**3.17(0.21)**	**3.19(0.25)**
FP rate (in %)	3.92(0.58)	10.96(2.96)	11.10(2.90)	11.45(3.05)	11.38(2.71)	10.64(2.05)
FN rate (in %)	5.09(0.55)	1.72(0.47)	1.65(0.44)	1.60(0.50)	1.66(0.45)	1.83(0.56)
	Cost-Ratio (r=0.1)					
test error (%)	**5.03(0.72)**	**2.41(0.34)**	**2.36(0.33)**	**2.33(0.34)**	**2.31(0.30)**	**2.39(0.29)**
FP rate (in %)	4.57(0.72)	13.33(2.42)	13.94(2.88)	15.17(4.04)	14.54(3.48)	13.94(2.43)
FN rate (in %)	5.07(0.75)	1.41(0.51)	1.30(0.53)	1.15(0.50)	1.18(0.57)	1.33(0.47)

Table 3.3 Comparison of standard SVM, cost-sensitive SVM and cost-sensitive U-SVM for real life MNIST data (using RBF kernel).

METHODS	standard SVM	cost-sensitive SVM	cost-sensitive U-SVM (digit 1)	cost-sensitive U-SVM (digit 3)	cost-sensitive U-SVM (digit 6)	cost-sensitive U-SVM (RA)
	Cost-Ratio (r=0.5)					
test error (%)	**1.34(0.28)**	**1.31(0.29)**	**1.23(0.37)**	**0.95(0.19)**	**1.15(0.34)**	**1.16(0.28)**
FP rate (in %)	1.10(0.73)	1.12(0.72)	0.96(0.66)	1.07(0.82)	0.89(0.74)	1.03(1.12)
FN rate (in %)	1.45(0.29)	1.41(0.3)	1.35(0.36)	0.89(0.27)	1.27(0.35)	1.23(0.27)
	Cost-Ratio (r=0.2)					
test error (%)	**1.59 (0.25)**	**1.45(0.20)**	**1.29(0.28)**	**0.97(0.31)**	**1.11(0.22)**	**1.17(0.28)**
FP rate (in %)	1.15(0.24)	3.19 (2.26)	3.43(2.69)	3.35(2.71)	2.64(2.27)	3.00(3.48)
FN rate (in %)	1.67(0.32)	1.13(0.44)	0.90(0.50)	0.53(0.39)	0.83(0.47)	0.84(0.54)
	Cost-Ratio (r=0.1)					
test error (%)	**1.50(0.24)**	**1.13(0.19)**	**1.11(0.17)**	**0.80(0.14)**	**0.90(0.22)**	**0.92(0.17)**
FP rate (in %)	1.31(1.47)	5.91(2.75)	6.57(3.27)	6.29(3.20)	5.24(2.54)	6.58(3.62)
FN rate (in %)	1.52(0.28)	0.69(0.37)	0.61(0.33)	0.30(0.19)	0.51(0.27)	0.41(0.27)

Table 3.4 Comparison of standard SVM, cost-sensitive SVM and cost-sensitive U-SVM on ISOLET ('B' vs. 'V' dataset) for different cost-ratios

METHODS	standard SVM	cost-sensitive SVM	cost-sensitive U-SVM (letter D)	cost-sensitive U-SVM (letter P)	cost-sensitive U-SVM (RA)
	Cost-Ratio (r=0.5)				
test error (in %)	**5.34(1.47)**	**5.21(1.23)**	**4.59(1.24)**	**4.33(0.82)**	**4.96(1.05)**
FP rate (in %)	9.36(2.31)	10.20(4.08)	10.32(3.88)	10.20(3.78)	9.52(3.40)
FN rate (in %)	3.32(1.61)	2.72(1.90)	1.72(1.21)	1.40(0.84)	2.68(1.85)
	Cost-Ratio (r=0.2)				
test error (in %)	**3.51(0.51)**	**3.42(0.42)**	**2.93(0.61)**	**2.77(0.52)**	**3.03(0.53)**
FP rate (in %)	11.68(3.20)	12.56(3.18)	12.6(3.83)	13.6(3.76)	11.96(2.59)
FN rate (in %)	1.88(0.98)	1.60(0.75)	1.00(0.74)	0.60(0.43)	1.24(0.74)
	Cost-Ratio (r=0.1)				
test error (in %)	**2.79(0.75)**	**2.70(0.65)**	**2.59(0.58)**	**1.78(0.42)**	**2.39(0.69)**
FP rate (in %)	12.24(3.81)	15.28(4.28)	14.88(4.07)	17.6(4.82)	14.6(3.93)
FN rate (in %)	1.84(0.76)	1.44(0.66)	1.36(0.60)	0.2(0.28)	0.48(0.45)

Table 3.5 Comparison of standard SVM, cost-sensitive SVM and cost-sensitive U-SVM on GTSRB ('50' vs. '80' dataset) for different cost-ratios

METHODS	standard SVM	cost-sensitive SVM	cost-sensitive U-SVM (sign 30)	cost-sensitive U-SVM (sign 60)	cost-sensitive U-SVM (RA)
	Cost-Ratio (r=0.5)				
test error (in %)	**9.82(0.83)**	**9.25(0.99)**	**6.75(1.09)**	**6.84(1.30)**	**8.91(0.60)**
FP rate (in %)	6.74(1.56)	8.84(3.77)	8.98(4.81)	9.78(5.53)	8.20(2.91)
FN rate (in %)	11.36(1.74)	9.46(1.65)	5.64(2.49)	5.38(1.73)	9.26(1.03)
	Cost-Ratio (r=0.2)				
test error (in %)	**9.27(1.25)**	**7.14(1.26)**	**5.88(0.82)**	**5.93(0.98)**	**6.91(1.13)**
FP rate (in %)	7.12(1.79)	19.75(4.97)	23.8(5.54)	26.07(3.57)	16.25(4.70)
FN rate (in %)	9.7(1.6)	4.63(2.06)	2.3(1.51)	1.9(0.91)	5.05(2.06)
	Cost-Ratio (r=0.1)				
test error (in %)	**9.44(1.70)**	**5.71(1.05)**	**4.74(1.15)**	**4.62(1.28)**	**4.77(0.75)**
FP rate (in %)	6.64(2.19)	45.02(18.69)	42.54(14.16)	44.98(14.27)	26.68(7.33)
FN rate (in %)	9.72(1.98)	1.78(1.32)	0.96(0.49)	0.58(0.45)	2.6(1.00)

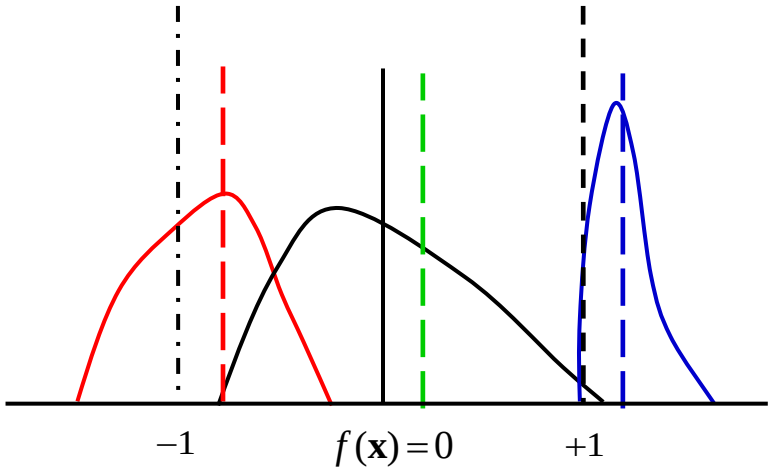


Figure 3.1: A schematic illustration of the histogram of projections onto normal w vector of cost-sensitive SVM decision boundary satisfying the practical conditions for the effectiveness of cost-sensitive U-SVM (when $r < 1$). Dashed red/blue lines indicate the training samples' class means. The average value of the two class means is shown in dashed green.

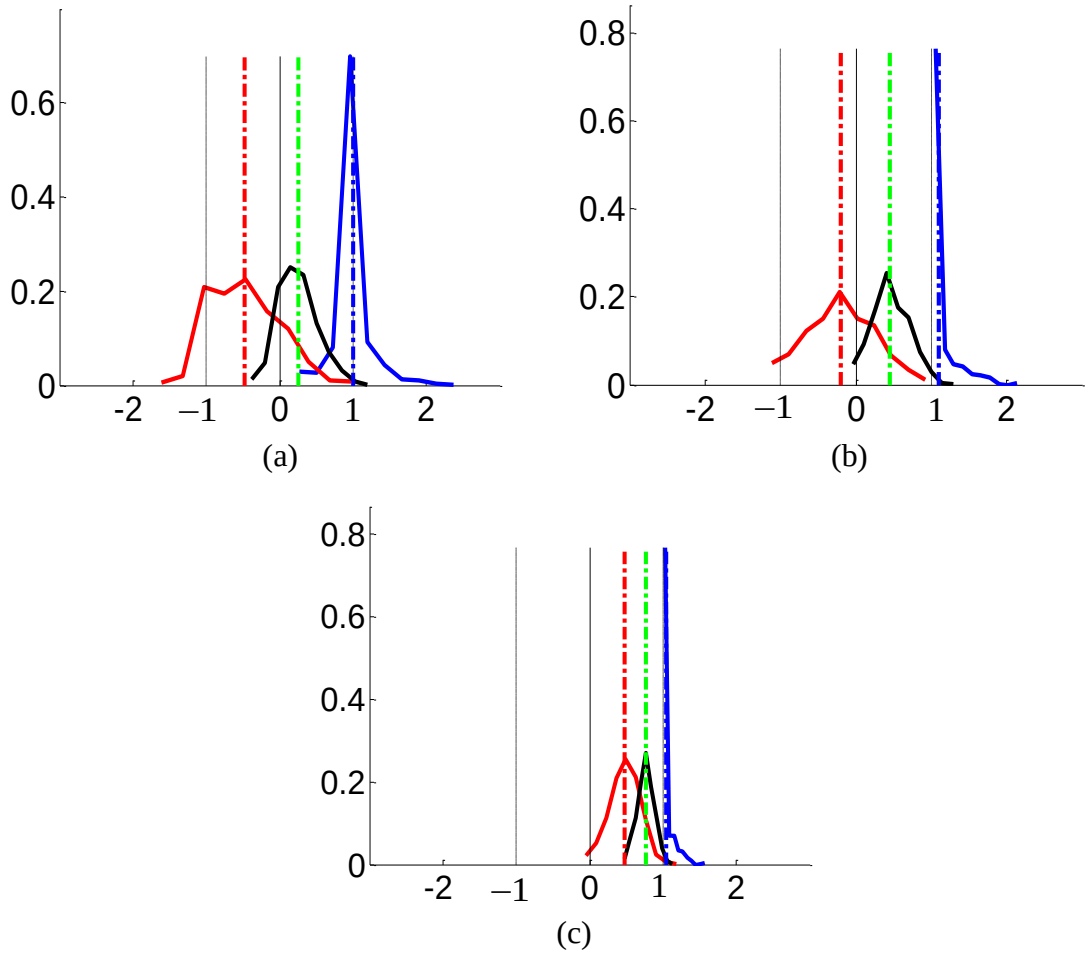


Figure 3.2: Univariate histogram of projections onto cost-sensitive SVM normal weight vector for synthetic hypercube data, for different cost-ratios (a) with r=0.5 ($C=2^{-6}$ and $C^*/C=2^{-4}$) (b) with r=0.2 ($C=2^{-5}$ and $C^*/C=2^{-8}$) (c) with r=0.1 ($C=2^{-5}$ and $C^*/C=2^{-5}$).

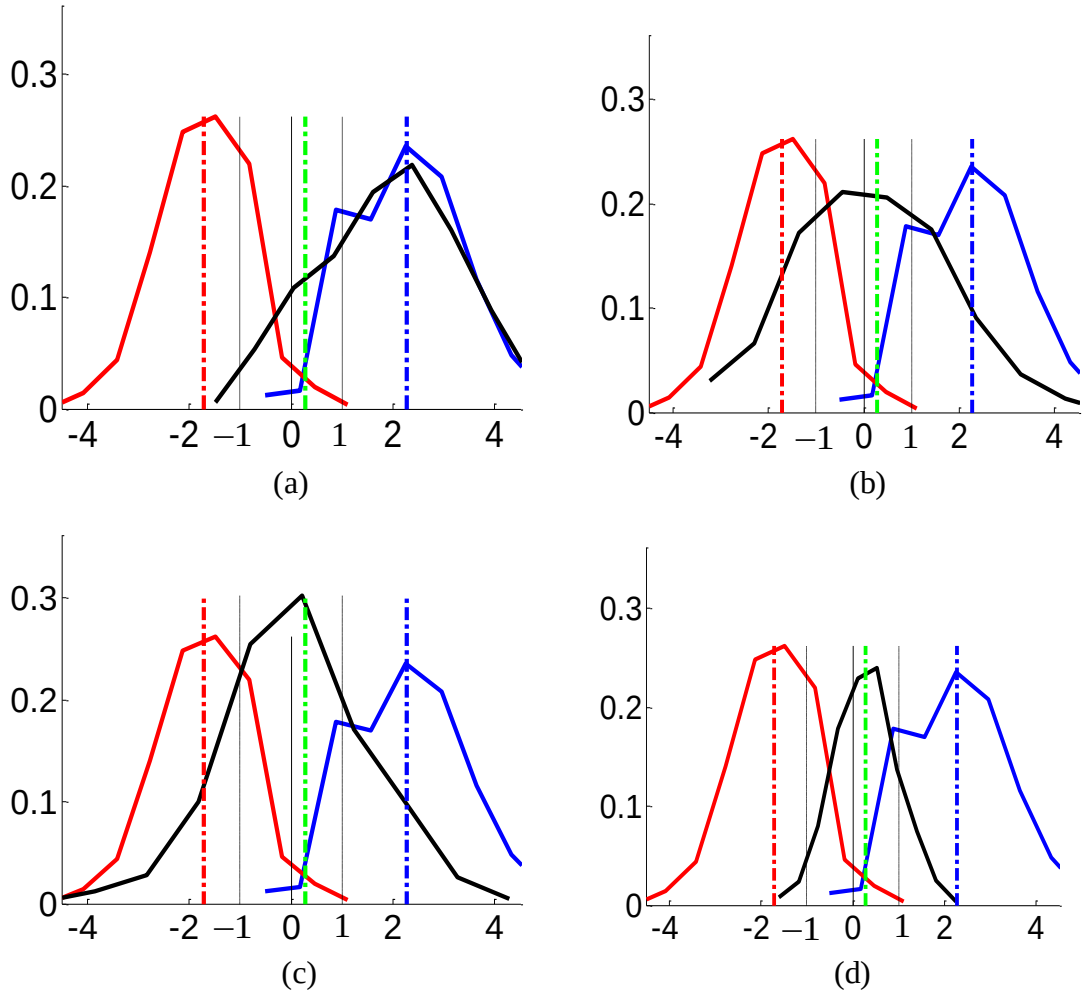


Figure 3.3: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-4}$) for MNIST data, for different types of Universa with r=0.5. Training set size ~1000 samples. Universum set size ~1000 samples. (a) Digit 1 Universum $C^*/C=2^{-9}$ (b) Digit 3 Universum $C^*/C=2^{-5}$ (c) Digit 6 Universum $C^*/C=2^{-8}$ (d) RA Universum $C^*/C=2^{-5}$.

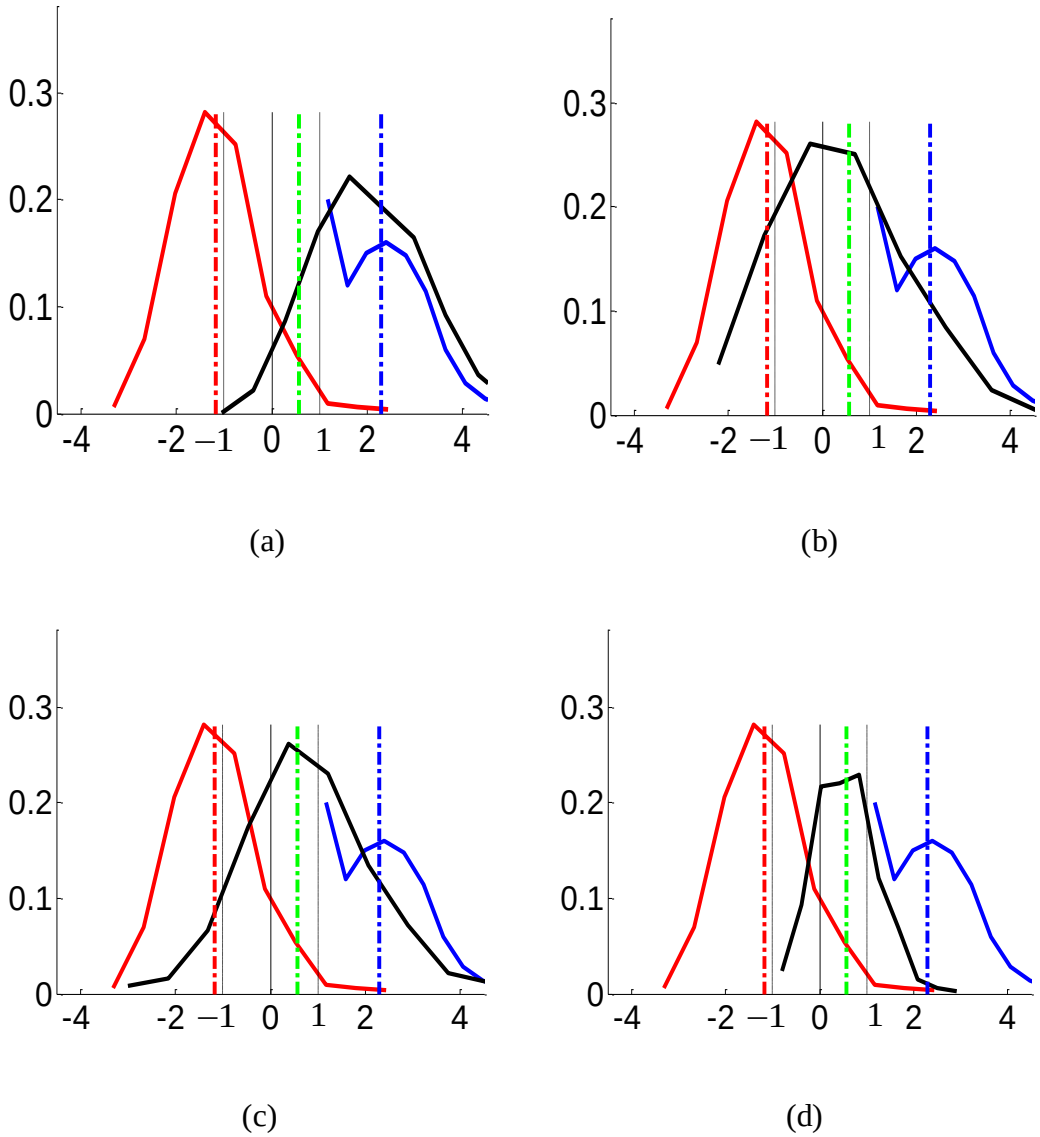


Figure 3.4: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-5}$) for MNIST data, for different types of Universa with r=0.1. Training set size ~1000 samples. Universum set size ~1000 samples. (a) Digit 1 Universum $C^*/C=2^{-20}$ (b) Digit 3 Universum $C^*/C=2^{-7}$ (c) Digit 6 Universum $C^*/C=2^{-10}$ (d) RA Universum $C^*/C=2^{-7}$.

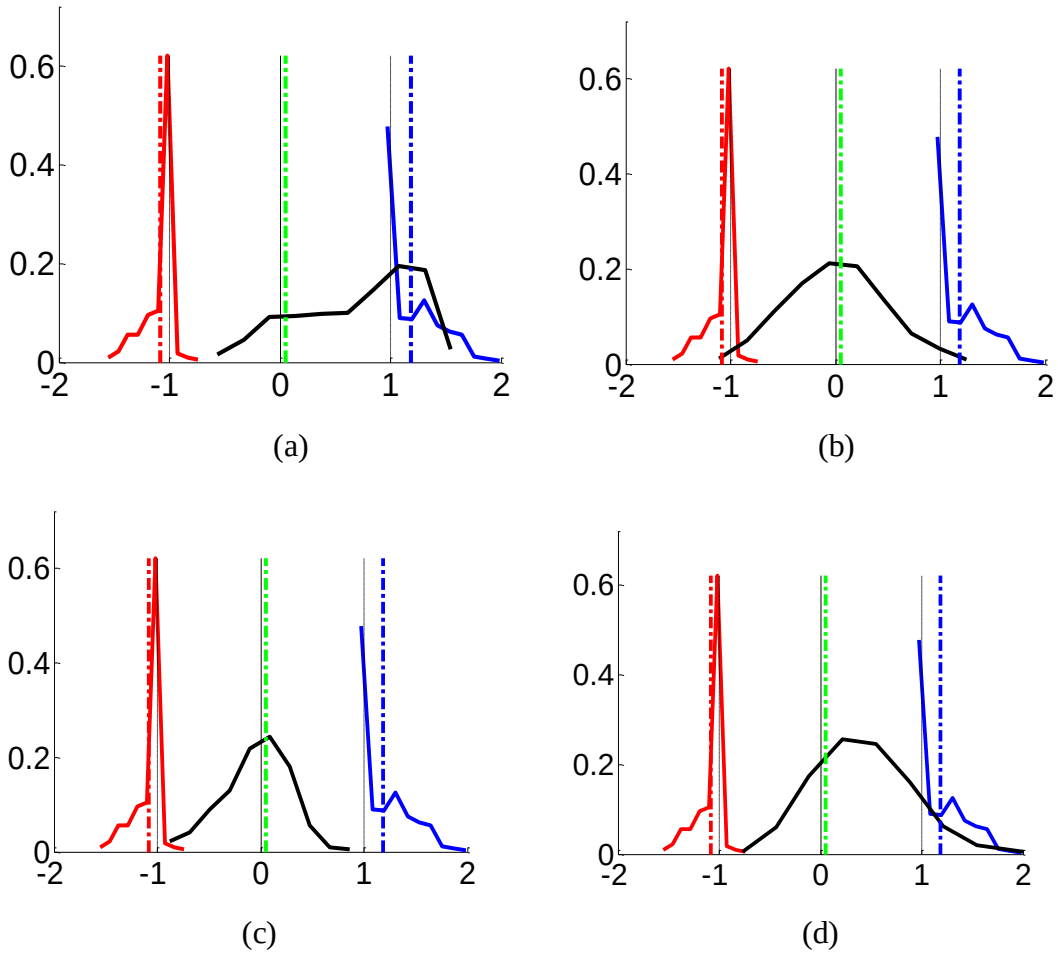


Figure 3.5: Univariate histogram of projections onto cost-sensitive SVM normal weight vector (C=2, $\gamma = 2^{-6}$) for MNIST data, for different types of Universa with r=0.5. Training set size ~1000 samples. Universum set size ~1000 samples. (a) Digit 1 Universum C*/C=2^{-4}. (b) Digit 3 Universum C*/C=2^{-2}. (c) Digit 6 Universum C*/C=2^{-2}. (d) RA Universum C*/C=2^{-1}.

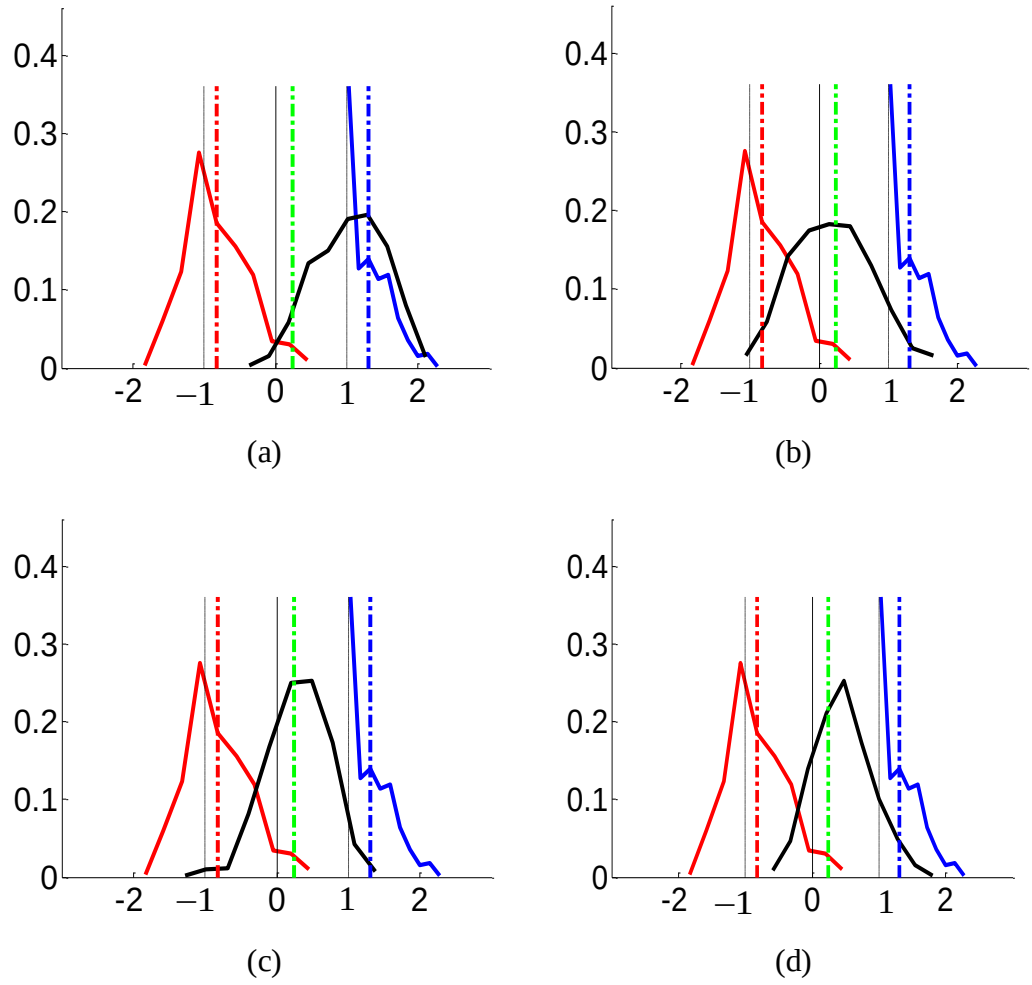


Figure 3.6: Univariate histogram of projections onto cost-sensitive SVM normal weight vector (C=2, $\gamma = 2^{-6}$) for MNIST data, for different types of Universa with r=0.1. Training set size ~1000 samples. Universum set size ~1000 samples. (a) Digit 1 Universum C*/C=2^{-4} (b) Digit 3 Universum C*/C=2^{-4} (c) Digit 6 Universum C*/C=2^{-4} (d) RA Universum C*/C=2^{-2}.

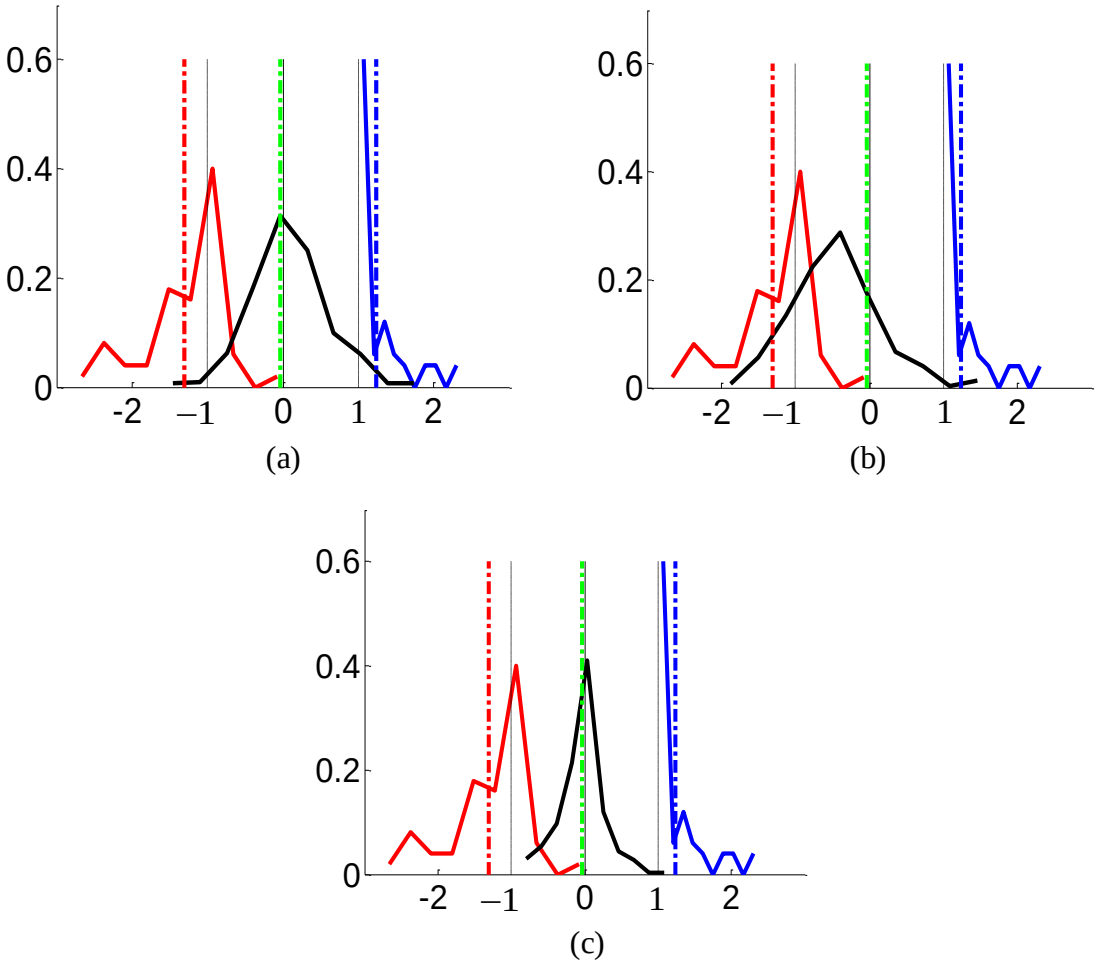


Figure 3.7: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-4}$) for ISOLET data, for different types of Universa for r=0.5. Training set size ~100 samples. Universum set size ~300 samples. (a) letter D Universum $C^*/C=2^{-4}$ (b) letter P Universum $C^*/C=2^{-5}$ (c) RA Universum $C^*/C=2^{-4}$.

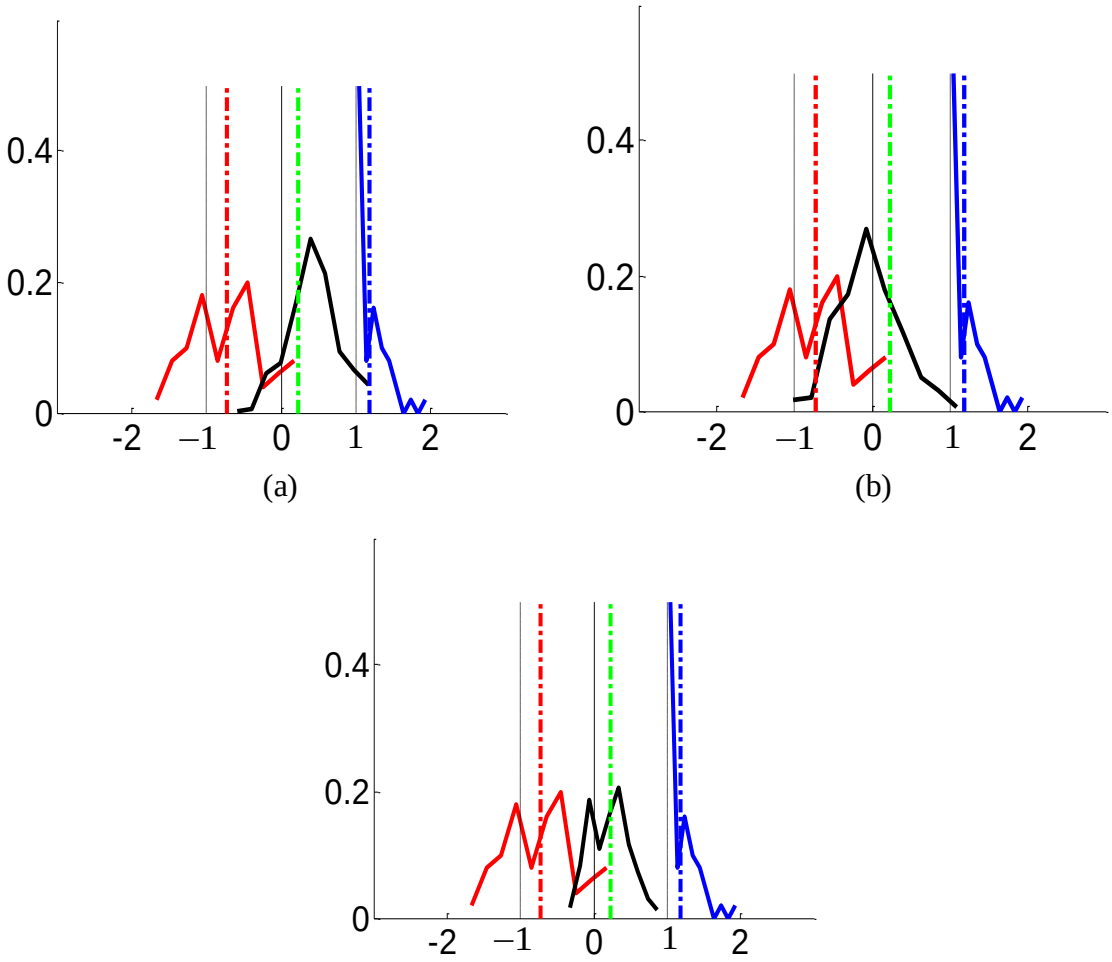


Figure 3.8: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-3}$) for ISOLET data, for different types of Universa for r = 0.1. Training set size ~100 samples. Universum set size ~300 samples. (a) letter D Universum $C^*/C=2^{-10}$ (b) letter P Universum $C^*/C=2^{-6}$ (c) RA Universum $C^*/C=2^{-5}$.

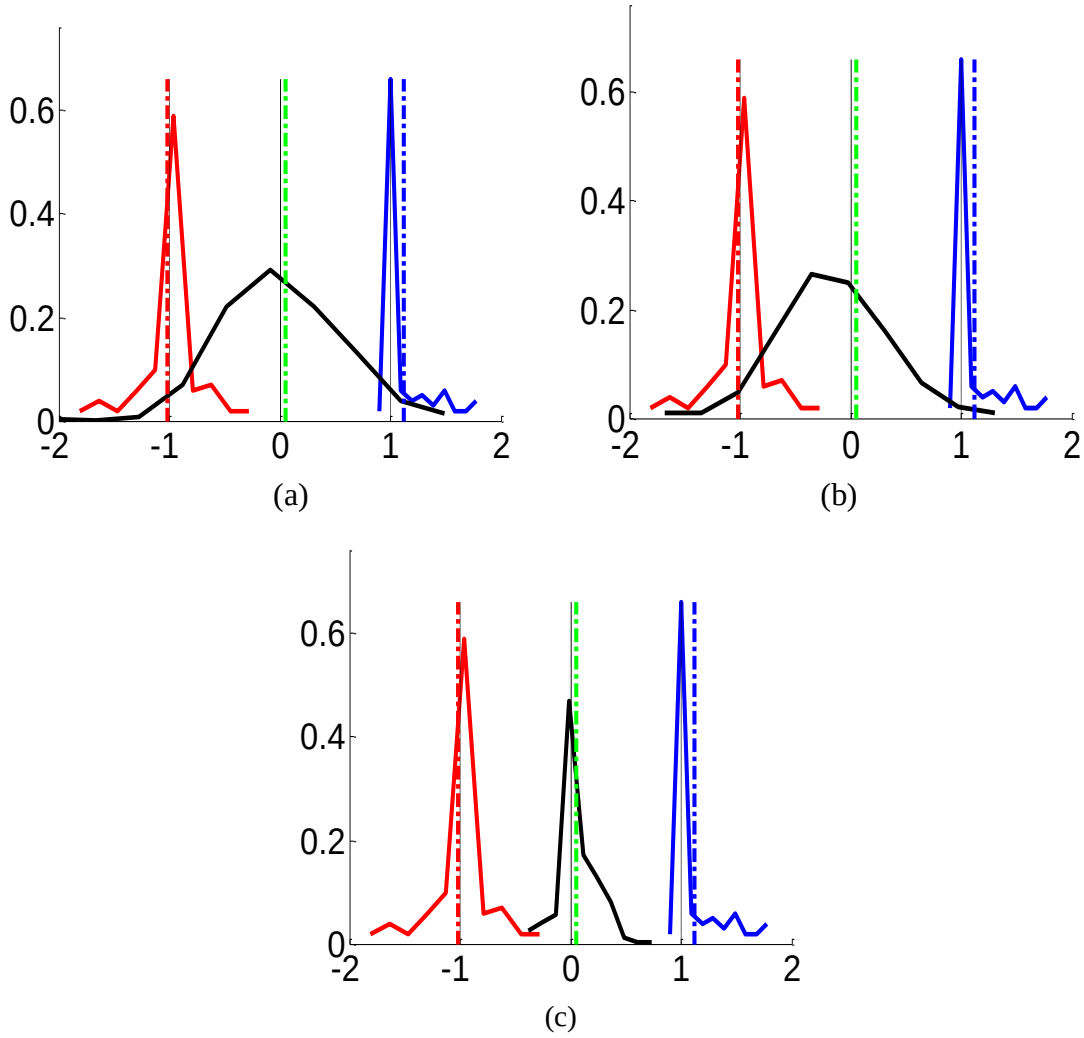
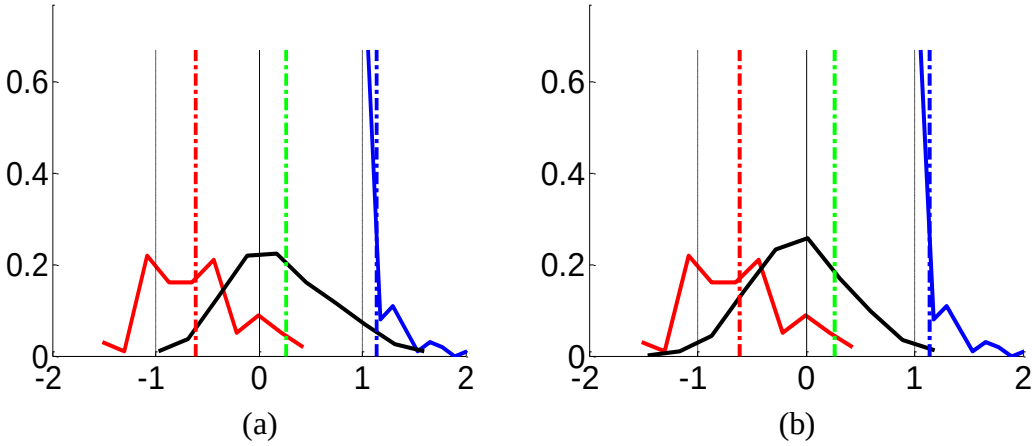
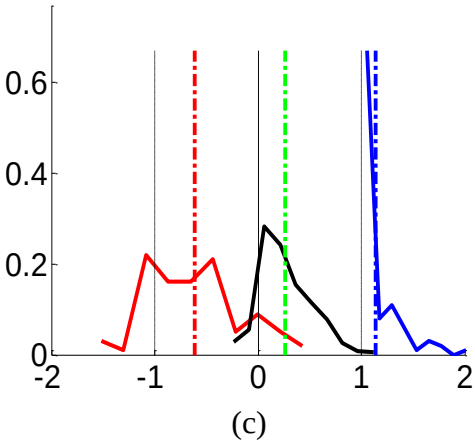


Figure 3.9: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-2}$) for GTSRB data, for different types of Universa for r=0.5. Training set size ~200 samples. Universum set size ~1000 samples. (a) sign '30' Universum $C^*/C=2^{-2}$ (b) sign '60' Universum $C^*/C=2^{-4}$ (c) RA Universum $C^*/C=2^{-5}$.



(a)

(b)



(c)

Figure 3.10: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-2}$) for GTSRB data, for different types of Universa for r=0.2. Training set size ~200 samples. Universum set size ~1000 samples. (a) sign '30' Universum $C^*/C=2^{-4}$ (b) sign '60' Universum $C^*/C=2^{-4}$ (c) RA Universum $C^*/C=2^{-7}$.

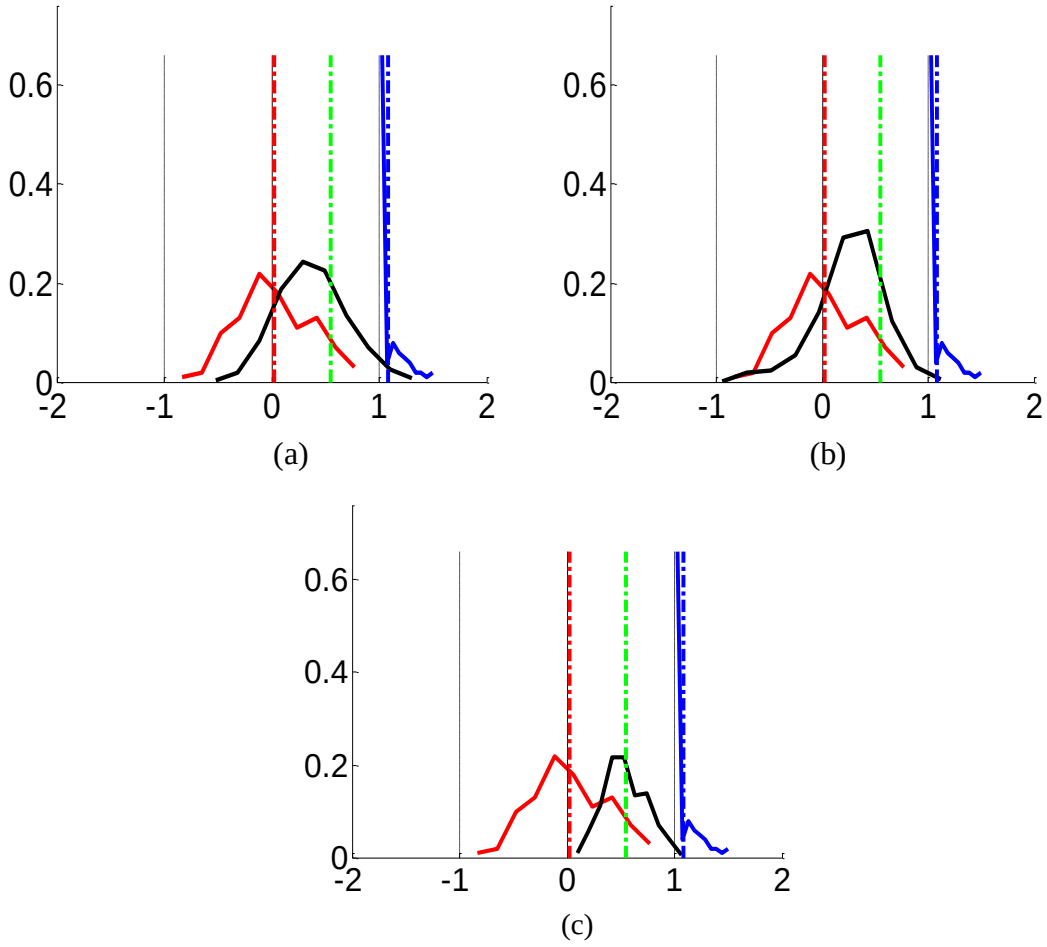


Figure 3.11: Univariate histogram of projections onto cost-sensitive SVM normal weight vector ($C=2^{-2}$) for GTSRB data, for different types of Universa for r=0.1. Training set size ~200 samples. Universum set size ~1000 samples. (a) sign '30' Universum $C^*/C=2^{-7}$ (b) sign '60' Universum $C^*/C=2^{-6}$ (c) RA Universum $C^*/C=2^{-6}$.

Chapter 4 Single-Class Universum Learning

4.1 Introduction

The idea of Universum learning or learning through contradiction (Vapnik, 1998; 2006) provides a formal mechanism for incorporating a priori knowledge about the application domain, in the form of additional (unlabeled) Universum samples. However, the implementation of Universum learning is *known only for classification setting*. It is not clear how to extend or modify this idea of learning through contradiction to other types of learning problems because the notion of 'contradiction' has been originally introduced for binary classification (Vapnik, 1998). This chapter extends the notion of Universum learning to *single-class learning* problems. For these problems, one can also expect to achieve improved generalization performance by incorporating a priori knowledge in the form of additional data samples from the same application domain. However, formalization of this idea requires significant effort. The main (conceptual) problem is that single-class model estimation (aka anomaly detection) represents unsupervised learning, where the notion of contradiction needs to be re-defined properly. This requires clear specification of the single-class learning itself.

Single-class learning problems are common in many real-life applications, such as object recognition, anomaly detection, fraud detection, document classification etc. (Schölkopf and Smola, 2002; Chandola et al, 2009; Manevitz and Yousef, 2002; Cherkassky and Mulier, 2007). This problem can be formalized as follows (Schölkopf

and Smola, 2002).

Problem Setting 1*: Given n samples drawn from an unknown probability distribution P(**x**), find a "simple" region $\mathcal{S}$ of input space, such that the probability that a test point drawn from P lies outside of $\mathcal{S}$ equals some fixed value (i.e., pre-specified False_Negative error rate).*

There are several known single class learning algorithms to solve this problem (Cherkassky and Mulier, 2007; Tan et al, 2006). One popular approach is the single-class SVM algorithm (Schölkopf and Smola, 2002), which estimates a binary function ***f(x)*** that takes the value +1 in a "*small*" region capturing most of the training data, and −1 elsewhere. Here the goal is to achieve a pre-specified false negative rate $FN\_rate = (1/n)\sum_{i=1}^{n} I(\mathbf{x}_i \notin \mathcal{S})$ on the training samples, which is expected to provide similar FN error rate for the future test samples drawn from the same distribution. As an illustration, consider the *handwritten digit recognition* problem, where the goal is to estimate a single-class decision rule for the images of the handwritten digit "0" in a 28x28 pixel space. The typical approach adopted is to estimate a single-class SVM model using the training samples of digit "0", which achieves a pre-specified user-defined *FN_rate*.

In many applications, single class SVM is used under a different problem setting, where in addition to test samples from distribution *P*, the *test* data also contains additional samples from a *different* distribution *Q*. Typically, samples from *P* are labeled as *normal (or positive) class* and those from *Q* are labeled as *abnormal class (aka negative)*. Under this setting, the goal is to estimate a single class model which minimizes

false positive rate *FP_rate* for a given *FN_rate*. Note that negative samples are not available (known) during training. For the digit recognition example, the normal class may represent samples of digit "0" and the abnormal class may constitute samples from other digits "1-9". This approach is implicitly adopted in many applications (Chandola et al, 2009; Manevitz and Yousef, 2002; Eskin et al, 2001; Eskin et al, 2002; Heller et al, 2003). Unfortunately, this approach is fundamentally flawed, in that *only positive samples* are available (known) during training a single-class model. So it is not possible, in principle, to achieve the goal of minimizing the FP error rate for test samples from a completely unknown distribution *Q*. This situation breeds many heuristic algorithms for single-class learning that exhibit 'superior' performance simply because an algorithm is a better match for specific data sets (Chandola et al, 2009; Manevitz and Yousef, 2002; Eskin et al, 2001; Eskin et al, 2002; Heller et al, 2003).

This chapter introduces a better setting for single class learning, where, in addition to positive training samples, one has 'Universum' samples from distribution *U* which is different from *P*. Universum samples belong to the same application domain and are defined in the same input space (**x**) as training and test samples. Therefore, Universum data *may* provide useful information about unknown *P* and *Q*, and so it can help minimize the number of FP errors for test samples from *Q*. For example, in the handwritten digit recognition problem, training data may include positive training samples of digit "0", Universum samples of handwritten letters (in the same 28x28 pixel space), and test samples of other digits "1-9" (~ negative class). The goal of learning is to estimate a single class model which minimizes FP_rate for a given FN_rate. Under this

setting, a single-class model is estimated from positive (normal) training samples *and* Universum samples. Even though Universum samples (letters) do not belong to the normal class, they reflect the style of handwriting, and can be possibly useful for minimizing FP error rate for 'other digits' from unknown distribution *Q*. This can be formalized as the following setting for single-class learning.

Problem Setting 2*: Given normal (positive) samples drawn from unknown distribution P(**x**), and Universum samples from unknown distribution U(**x**), find a "simple" region $\mathcal{S}$ of input space that provides pre-specified FN error rate and minimum FP error rate for future test samples. It is assumed that test samples are generated from some mixture of P(**x**) and Q(**x**), aka positive (normal) and negative (abnormal) class distributions.*

Under this setting, additional samples from *U(**x**)* contain additional a priori knowledge about application domain, similar to Universum learning originally introduced for binary classification setting (Vapnik, 1998; 2006). However, the idea of 'learning through contradiction' (Vapnik, 1998; 2006) cannot be easily extended to single-class learning, where negative samples are not available for training (model estimation). This chapter shows how to incorporate Universum samples into SVM-like formulation for single class learning, in order to solve *Problem Setting 2*.

The chapter is organized as follows. Section 4.2 provides the background on single-class SVM developed in (Schölkopf and Smola, 2002). Section 4.3 describes the proposed single-class U-SVM formulation, as a constructive solution for single-class learning (under Setting 2). Implementation of single-class U-SVM is provided in section

4.4. Section 4.5 presents empirical results for single-class U-SVM. Finally, conclusions are presented in Section 4.6.

4.2 Single-Class SVM

Single-class SVM formulation (Schölkopf and Smola, 2002) was proposed for solving Problem Setting 1. For improved readability, we show only linear SVM parameterization.

Given, training data from the *"normal class"* $\{\mathbf{x}_i\}_{i=1}^{n}$

Solve the following optimization problem,

$$\min_{\mathbf{w},\rho,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \tag{4.1}$$

s.t. $(\mathbf{w}\cdot\mathbf{x}_i) \geq \rho\text{-}\xi_i$, $\xi_i \geq 0$, $i = 1 \text{ to } n$

Finally, predict on future test data as $D(\mathbf{x}_i, \mathbf{w}, \rho) = \begin{cases} +1 \text{ ("normal")} & \text{if } (\mathbf{w}\cdot\mathbf{x}_i) \geq \rho \\ -1 \text{ ("abnormal")} & \text{otherwise} \end{cases}$

It can be readily extended to nonlinear SVM via kernels (Schölkopf and Smola, 2002, Cherkassky and Mulier, 2007). Under this approach, a large margin (ρ) hyperplane, characterized by a margin size $\rho/\|\mathbf{w}\|$, is used to separate the training samples from the origin (as shown in Fig. 4.1). Samples lying outside this margin are linearly penalized using the slack variables ξ_i. The tradeoff between margin size and margin errors is controlled by the user defined parameter $\nu \in [0,1]$. This parameter ν also acts as the lower bound on the fraction of training points that are support vectors and controls the FN_rate for training samples. This indirectly controls the FN_rate for test samples (from

the same distribution). Typically, a small value of ν is likely to provide a small FN_rate on the training as well as the future test samples. Selecting this parameter is application dependent and is set (by a user) based on application domain knowledge (Cherkassky and Mulier, 2007). Technically, fixed FN_rate for test data can be achieved using independent validation dataset (or via resampling).

Schölkopf and Smola (2002), proposed to map a single-class SVM problem onto an 'equivalent' binary SVM classification problem, as shown in Fig. 4.2(a), (b). Their proposition 8.2 shows how the margin of an equivalent binary SVM classifier is related to the single-class SVM decision boundary (shown in Fig 4.2c):

Proposition 8.2: *Suppose* $(\mathbf{w}, b = 0)$ *parameterizes the optimal separating hyperplane passing through the origin for a labeled data set* $\{(\mathbf{x}_1, y_1 = +1), ..., (\mathbf{x}_n, y_n = 1), (-\mathbf{x}_1, y_{n+1} = -1), ..., (-\mathbf{x}_n, y_{2n} = -1)\}$ *aligned such that* $(\mathbf{w} \cdot \mathbf{x}_i)$ *is positive for* $y_i = +1$. *Suppose, moreover, that* $\rho / \|\mathbf{w}\|$ *is the margin of the optimal hyperplane. Then* $(\mathbf{w}, \rho)$ *parameterizes the supporting hyperplane for the unlabeled data set* $\{\mathbf{x}_i\}_{i=1}^{n}$.

Proof: See (Schölkopf and Smola, 2002).

This connection yields the following Algorithm 4.1 for solving single-class SVM via binary SVM classification. (A schematic representation of this algorithm is shown in Fig. 4.2).

Algorithm 4.1 Solving single class SVM problem using a binary SVM classifier (see Fig. 4.2)

1. Given, training data from the *"normal class"* $\{\mathbf{x}_i\}_{i=1}^{n}$. Reflect the data about the origin and label them as shown (see Fig. 4.2a),

$$\mathbf{x}_i = \mathbf{x}_i \quad \text{with } y_i = +1, \quad \text{for } i = 1 \text{ to } n$$

$$= -\mathbf{x}_i \quad \text{with } y_i = -1, \quad \text{for } i = n+1 \text{ to } 2n$$

2. Solve the binary SVM problem

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{2n}\xi_i \tag{4.2}$$

$$\text{s.t} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1-\xi_i \qquad \xi_i \geq 0 \qquad i = 1 \text{ to } n$$

3. The single class SVM decision rule is given as:

$$D(\mathbf{x}_i, \mathbf{w}) = \begin{cases} +1 \text{ ("normal") ; if } (\mathbf{w}\cdot\mathbf{x}_i) \geq 1 \\ -1 \text{ ("abnormal") ; otherwise} \end{cases}$$

Here the parameter C is equivalent to the parameter ν, and controls the *FN_rate* of the training as well as the future test samples. Note that, for improved readability, we show only linear parameterization in Algorithm 4.1. However (4.2) can be generalized to the nonlinear case by solving the problem in the dual form and using the kernel $\mathbf{K} = \begin{bmatrix} \mathbf{K}_T & -\mathbf{K}_T \\ -\mathbf{K}_T & \mathbf{K}_T \end{bmatrix}$, where $\mathbf{K}_T$ is the kernel for the training samples. This modified

kernel $\mathbf{K}$ captures the effect of reflecting the training samples about the origin in the kernel space.

Note that Algorithm 4.1 effectively shows how to solve single-class Setting 1 via binary SVM classification. This connection enables application of existing SVM classification software for single-class Setting 1. In the next section, we use this connection to introduce the new single-class Universum SVM formulation.

4.3 Single-Class U-SVM

Next, we introduce our new formulation called the single-class Universum support vector machine (U-SVM) used to solve Problem Setting 2. Under this setting we are given training samples from normal class and a set of examples from the Universum. The Universum contains data that belongs to the same application domain as the training data, but these samples are known not to follow the same distribution as the "normal" class. Note however that the Universum samples *may or may not* follow the same distribution as the abnormal class. These Universum samples are incorporated into single-class learning as explained next. Let us assume that the training samples are linearly separable using two large margin hyperplanes $\mathcal{H}_1$ and $\mathcal{H}_2$ with the same margin size. Then the Universum samples can fall on either side of the decision boundary (see Fig. 4.3a). Note that, we should favor hyperplane models ($\mathcal{H}_2$) where the Universum fall on the wrong side of the decision boundary (i.e., they should not be classified as the normal class). Such Universum samples are called contradictions, because they are falsified by the

model (i.e., have nonzero slack variables). Thus, Universum learning implements a trade-off between explaining training samples from the normal class (using large margin hyperplanes) and maximizing the number of contradictions (on the Universum).

The quadratic optimization formulation for implementing an SVM-style inference through contradictions is introduced next. For training samples, we use the standard single-class SVM soft-margin loss with slack variables ξ_i. The universum samples $(\mathbf{x}_j^*)$ are penalized using a hinge loss on the universum samples $L_\varepsilon(f(\mathbf{x},\mathbf{w})) = \max((\mathbf{w}\cdot\mathbf{x}) - \varepsilon, 0)$, with $\varepsilon > 0$ (as shown in Fig. 4.4a). This loss function forces the Universum to lie far away from the decision boundary. That is, the universum samples with $(\mathbf{w}\cdot\mathbf{x}) \geq \varepsilon$ are linearly penalized using the slack variables ξ_j^*. The proposed single-class U-SVM formulation is shown next:

(samples from "*normal*" class) (samples from "*universum*")

$$\min_{\mathbf{w},\boldsymbol{\xi}} \quad \boxed{\frac{1}{2}\|\mathbf{w}\|^2 + \hat{C}\sum_{i=1}^{n}\xi_i} \quad + \quad \boxed{C^*\sum_{j=1}^{m}\xi_j^*} \qquad (4.3)$$

$$\text{s.t.} \quad \boxed{\mathbf{w}\cdot\mathbf{x}_i \geq 1\text{-}\xi_i} \qquad \boxed{\mathbf{w}\cdot\mathbf{x}_j^* \leq \varepsilon + \xi_j^*}$$

$$\boxed{\xi_i \geq 0,\ i = 1 \text{ to } n} \qquad \boxed{\xi_j^* \geq 0,\ j = 1 \text{ to m}}$$

Here, the user-defined parameters $\hat{C}, C^* \geq 0$ control the tradeoff between minimizing the FN_rate on training samples and maximizing the number of contradictions on the universum samples.

An alternative assumption is that the Universum samples *do not* follow the same distribution as the normal *as well as the abnormal class*. Under such scenarios the single-

class U-SVM formulation (4.3) needs to be changed. For such settings, the Universum samples are incorporated into single-class learning as explained next. Let us assume that the training samples are linearly separable using two large margin hyperplanes $\mathcal{H}_1$ and $\mathcal{H}_2$ with the same margin size (see Fig. 4.3b). Here, the Universum samples follow a distribution different than both the normal as well as the unseen abnormal class. Note that, here we should favor hyperplane models ($\mathcal{H}_1$) which are closer to the universum samples; and hence far from the unseen abnormal class.

The quadratic optimization formulation for implementing an SVM-style inference through contradictions for this case is introduced next. For training samples, we use the standard single-class SVM soft-margin loss with slack variables ξ_i. The universum samples $(\mathbf{x}_j^*)$ are penalized using an ε- insensitive loss on the universum samples with $\varepsilon > 0$ (as shown in Fig. 4.4b). This loss function forces the Universum to lie close to the decision boundary within an ε-insensitive zone. Here, the universum samples with $|\mathbf{w} \cdot \mathbf{x} - 1| \geq \varepsilon$ are linearly penalized using the slack variables ξ_j^*. Such Universum samples are called contradictions, because they are falsified by the model (i.e., have nonzero slack variables). The proposed single-class U-SVM formulation is shown next:

(samples from "*normal*" class) (samples from "*universum*")

$$\min_{\mathbf{w},\boldsymbol{\xi}} \quad \boxed{\frac{1}{2}\|\mathbf{w}\|^2 + \hat{C}\sum_{i=1}^{n}\xi_i} \quad + \quad \boxed{C^*\sum_{j=1}^{m}\xi_j^*}$$

$$\text{s.t.} \quad \boxed{\mathbf{w}\cdot\mathbf{x}_i \geq 1\text{-}\xi_i} \qquad \boxed{|\mathbf{w}\cdot\mathbf{x}_j^* - 1| \leq \varepsilon + \xi_j^*}$$

$$\boxed{\xi_i \geq 0,\ i = 1 \text{ to } n} \qquad \boxed{\xi_j^* \geq 0,\ j = 1 \text{ to m}} \tag{4.4}$$

Here, the user-defined parameters $\hat{C}, C^* \geq 0$ control the tradeoff between minimizing the FN_rate on training samples and maximizing the number of contradictions on the universum samples. In this thesis we limit ourselves to the first formulation (4.3). Analysis of the single-class U-SVM formulation (4.4) has been left for future research.

4.4 Implementation of the Single-Class U-SVM Formulation

Formulation (4.3) can be solved via an equivalent binary classification formulation, effectively following the same steps as in Algorithm 4.1 (shown in Fig. 4.2). This leads to the following Algorithm 4.2 for solving (4.3).

Algorithm 4.2: Solving single class U-SVM problem using a binary U-SVM classifier

---

1. Given, training data from the *"normal class"* $\{\mathbf{x}_i\}_{i=1}^{n}$ and additional universum samples $\{\mathbf{x}_j^*\}_{i=1}^{m}$. Reflect the training data and label them as shown (see Fig. 4.2a),

   $\mathbf{x}_i = \mathbf{x}_i$ with $y_i = +1$, for $i = 1$ to n

   $= -\mathbf{x}_i$ with $y_i = -1$, for $i = n+1$ to $2n$.

   Note, we do not reflect the universum samples.

2. Solve the binary U-SVM problem (Vapnik, 2006; Weston et al, 2006),

$$\min_{\mathbf{w},b,\xi} \quad \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{2n}\xi_i}_{\text{(samples from "normal" class)}} + \underbrace{C^*\sum_{j=1}^{m}\xi_j^*}_{\text{(samples from "universum")}} \qquad (4.5)$$

$$\text{s.t.} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \; i = 1 \text{ to } 2n$$

$$\left|(\mathbf{w}\cdot\mathbf{x}_j^*) + b\right| \leq \varepsilon + \xi_j^*, \quad \xi_j^* \geq 0, \; j = 1,\ldots,m$$

3. Finally, predict on the future test data $D(\mathbf{x}_i, \mathbf{w}) = \begin{cases} +1 \text{ ("normal")} & \text{if } (\mathbf{w}\cdot\mathbf{x}_i) \geq 1 \\ -1 \text{ ("abnormal")} & \text{otherwise} \end{cases}$

The solution to the optimization problem (4.3) defines the large margin hyperplane $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) - 1 = 0$ that incorporates a priori knowledge (i.e., Universum samples) into the final single-class model. Following the same arguments as in proposition 8.2 (Schölkopf and Smola, 2002), we have b = 0. Hence (4.3) is equivalent to solving (4.4) with $\hat{C} = 2C$, except for an additional constraint $-\mathbf{w} \cdot \mathbf{x}_j^* \leq \varepsilon + \xi_j^*$ on the universum samples. Note that for Universum samples, constraint $-\mathbf{w} \cdot \mathbf{x}_j^* \leq \varepsilon + \xi_j^*$ acts as an inactive constraint and does not affect the solution. The computational cost for solving this single-class U-SVM problem is the same as solving the standard binary U-SVM with *2n* training samples and *m* universum samples. This is equivalent to solving the standard binary SVM problem with 2(n+m) samples (Weston et al, 2006).

4.5 Practical Conditions for the Effectiveness of Single-Class U-SVM

As before, there are *two design factors* necessary for a successful practical application of single class U-SVM.

- *Model Selection*: which becomes rather difficult because the kernelized U-SVM has 4 tunable parameters: C, C^*, kernel parameter and ε (in contrast, standard SVM has only two tuning parameters).
- generalization performance of U-SVM may be also affected by a bad choice of the Universum data.

Hence we adopt a similar strategy as 2.4 (a)-(d) for judging the effectiveness of a given Universum. However, now the univariate histogram is generated by projecting the

training and universum samples onto the normal direction vector of the single-class SVM hyperplane.

STRATEGY TO ANALYZE THE EFFECTIVENESS OF SINGLE-CLASS U-SVM	
4.5a.	perform model selection for single-class SVM to ensure given FN rate on independent validation set (or via resampling of training data).
4.5b.	generate low-dimensional representation of training data by projecting it onto the normal direction vector of one class SVM hyperplane estimated in (4.5 a) (see Fig. 4.5).
4.5c.	project the Universum data onto the normal direction vector of the one class SVM hyperplane (see Fig. 4.6a).
4.5d.	analyze the histogram of projected Universum data in relation to projected training data (see Fig. 4.6b).

The benefits of this strategy are two- fold. First, it simplifies the characterization of good Universum data. Specifically, based on the statistical properties of the projected Universum data relative to training data (in step. 4.5d), we can formulate the conditions on whether using this Universum will improve performance of the single-class SVM estimated in step 4.5a, for a given false negative rate. The practical conditions for the effectiveness of one-class U-SVM are provided below and illustrated in Fig. 4.7.

PRACTICAL CONDITIONS FOR EFFECTIVENESS OF SINGLE CLASS U-SVM	
C1.	The histogram of projections of the training data is well separable from origin and lie outside the decision boundary given as $f(\mathbf{x}) = 1$.

The histogram of projections of the Universum data:

C2.	It has wide distribution between the one-class decision boundary ($f(\mathbf{x}) = 1$) and $f(\mathbf{x}) = 0$ in the projection space.

Further, as before this leads to the following two-step strategy for model selection (parameter tuning) for the single-class U-SVM:

a Perform model selection (C and kernel parameter) for single-class SVM to ensure given FN rate on independent validation set (or via resampling of training data).

b Perform model selection for the ratio C*/C in (4.3) while keeping C and kernel parameters fixed (as in (a)). This is achieved by choosing the maximum value of C*/C providing fixed (pre-specified) FN error rate on an independent validation set (of positive samples). Parameter ε is usually pre-set to a small value and does not require tuning. For this chapter we set $\varepsilon = 0$.

Further, the performance of the single-class U-SVM may also depend on the number of universum samples used. For simplicity, in this thesis we keep it equal to the number of training samples. Next, we provide empirical results to show the effectiveness of the proposed single-class U-SVM over single-class SVM under Problem Setting 2.

4.6 Empirical Results for Single-Class U-SVM

4.4.1 MNIST

For our first set of experiments we use the real-life handwritten digits recognition MNIST data (**http://www.cs.nyu.edu/~roweis/data.html**). The goal is to build a single-class classifier for digit "0", where each sample is represented as a real-valued vector of size 28x28=784. For illustration we show the results for handwritten digits "1" and "2" as universum. We test our estimated single-class models under two distinct scenarios. In the first case, the "abnormal" class constitutes of the handwritten digits "3 to 9". Here, the samples of the abnormal class follow a different distribution than the normal class as well as the universum samples. For the second case, the "abnormal" class contains handwritten digits "1" or "2". In this case, the abnormal class follows the same distribution as the universum. The experimental setting used for this example follows next,

- No. of training/validation samples = 1000. (digit "0"). (The validation set contains independent samples of digit "0". This validation set is used to select the C value for single-class SVM. Further it is also used to select the C*/C (ratio) parameter specific to single-class U-SVM, which provides the same FN_rate on the validation set, as the single-class SVM).
- No. of additional Universum samples = 1000. (digit "1" and "2").
- No. of test samples: (we use all the samples available in the separate test set (**http://www.cs.nyu.edu/~roweis/data.html**)
  - *normal* class (from digit "0") with 980 samples.

- *abnormal* class (*for case 1:* 6853 samples from digits 3 to 9).
- *abnormal* class (*for case 2*: 1135 samples from digit "1" ; 1032 samples from digit "2").

We use linear SVM parameterization which is appropriate for this (sparse) data set. We provide our results for two different model parameters $\nu = 2^{-8}, 2^{-4}$; characterizing high and low FN_rates on the training / test samples from the "normal" class. Table 4.1 shows performance comparison for the single-class SVM vs. single-class U-SVM. The table shows the average value of the FN_rate and FP_rate (in percent) for 10 random selection of the training/validation set. The test set is kept fixed. The standard deviation of the FN_rate and FP_rate are shown in parenthesis. The typical histograms of projections for training data along with the Universum data are shown in Fig. 4.8-4.9. In all figures the training samples are shown in blue. The projection of the universum samples are shown in black.

For cases with high values of $\nu = 2^{-4}$ (characterizing high FN_rate), the training data is not well-separable (see Fig. 4.9). For such cases, the original motivation for Universum learning, to stabilize selection of a large margin hyperplane does not work. This is also confirmed from the results shown in Table 4.1.

However, for $\nu = 2^{-8}$ the histograms of projections in Fig 4.8 have the following characteristics,

– the training samples are well-separable from the origin and lie outside the decision boundary.
– digit "1" has well spread projections between the $f(\mathbf{x}) = 0$ and $f(\mathbf{x}) = 1$ boundaries.

– digit “2” universum lies mostly outside the $f(\mathbf{x})=1$ decision boundary.

Practical conditions (C1)-(C2) indicate that for the given well-separable training samples; digit ‘1’ is a better choice for Universum samples. The digit ‘2’ is highly biased towards the ‘+’ve class (digit ‘0’); hence is not likely to provide any improvement.

This is also confirmed from the results shown in Table 4.1. For this case with high-separability of the training samples (i.e., for $\nu = 2^{-8}$ with low FN_rate), the single class U-SVM provides a significant improvement using digit “1”. Such an improvement is not seen with digit “2” as universum. This can also be (intuitively) explained by noting visual similarity between digits “2” and “0”. Hence, digit “2” is not a good contradiction for digit “0”.

4.4.2 Reuters 21578 v1.0

Our next set of results uses the real-life Reuters-21578 v1.0 data **(http://www.daviddlewis.com/resources/testcollections /reuters21578/)**. It consists of 21,578 news stories that appeared in the Reuters newswire in 1987, which are classified according to 135 thematic categories mostly concerning business and economy. Here we use the subset R90 of this collection and the standard ModApte’ split to define the documents used as training and testing examples (see Correa and Ludermir, 2008). We use the preprocessed term-frequency encoded data using 5180 selected words already available in (Correa and Ludermir, 2008). The goal is to build a single-class classifier for the category “crude”. We show the results for two types of Universa: “money-fx” and “trade”. As before, we consider two extreme cases for the “abnormal class”. For the first

case, the abnormal class consists of all samples from the 90 other categories; except the categories "crude" (normal class), "money-fx" or "trade" (universum). For the second case, the "abnormal" class consists only of the unseen samples from the universum (i.e. "money-fx" or "trade"). The experimental setting is detailed below,

- No. of training/ validation samples = 195. (samples from category "crude").
- No. of additional Universum samples = 200. (samples from "money-fx", "trade").
- No. of test samples: we use all the samples available in the separate test set (Correa and Ludermir, 2008)
    - normal class (from category "crude") with 189 samples,
    - abnormal class (case 1: 2539 samples from 90 other categories except "crude", money-fx", "trade"), abnormal class (case 2: 517 samples from "money-fx"; 286 samples from "trade").

We use linear SVM parameterization which is appropriate for this (sparse) data set. The typical histograms of projections for training data along with the Universum data are shown in Fig. 4.10. For cases with high values of $\nu = 2^{-4}$ (characterizing high FN_rate), the training data is not well-separable (see Fig. 4.11); and hence does not provide any improvement. However, for $\nu = 2^{-8}$ the histograms of projections in Fig 4.10 have the following characteristics,

- the training samples are well-separable from the origin and lie outside the decision boundary.

– "money-fx" has well spread projections between the $f(\mathbf{x}) = 0$ and $f(\mathbf{x}) = 1$ boundaries.
– "trade" universum lies mostly outside the $f(\mathbf{x}) = 1$ decision boundary.

Based on the practical conditions (C1)-(C2) "money-fx" is a better choice for Universum samples. The "trade" universum is highly biased towards the '+'ve class; and is not likely to provide any improvement.

This is also confirmed from the results shown in Table 4.2. For this case with high-separability of the training samples (i.e., for $\nu = 2^{-4}$ with low FN_rate), the single class U-SVM provides a significant improvement using "money-fx". Such an improvement is not seen with "trade" as universum.

4.4.3 USPS

Finally, for our final set of experiments we use the real-life handwritten digits recognition USPS data (Schölkopf and Smola, 2002). The goal is to build a single-class classifier for digit "0", where each sample is represented as a real-valued vector of size 16x16=256. In this case however, we use the handwritten digits "3" and "4" as universum. Here, the "normal class" constitutes of the samples from digit "0". We consider two extreme scenarios for the "abnormal" class. In the first case, the "abnormal" class constitutes of the handwritten digits of all the other digits, i.e., "1, 2, 5, 6, 7, 8 and 9" and follows a totally different distribution than the universum samples. For the second case, the "abnormal" class constitutes only of unseen handwritten digits "3" and "4" and follows the same distribution as the universum samples.

The experimental setting used for this example follows,

- No. of training/validation samples = 500. (samples from digit “0”).
- No. of additional Universum samples = 500. (samples from digit “3”, “4”).
- No. of test samples: we use all the samples available in the separate test set
  - normal class (from category “0”) with 359 samples,
  - abnormal class (case 1: 1282 samples from digits “1,2,5,6,7,8,9”),
  - abnormal class (case 2: samples from digit “3”; samples from digit “4”)

Following (Schölkopf and Smola, 2002) we use an RBF kernel of the form $K(\mathbf{x},\mathbf{x}') = \exp\left(-\gamma\|\mathbf{x}-\mathbf{x}'\|^2\right)$ with $\gamma = 2^{-7}$. The typical histograms of projections for training data along with the Universum data are shown in Fig. 4.12-4.13. For this data the histograms of projections have the following characteristics,

- the training samples are well-separable from the origin and lie outside the decision boundary.
- digit “3” is biased towards and lies outside $f(\mathbf{x}) = 1$ boundaries.
- digit “4” lies within the $f(\mathbf{x}) = 0$ and $f(\mathbf{x}) = 1$ boundaries.

Based on the practical conditions (C1)-(C2) digit “4” is a better choice for Universum samples.

This is also confirmed from the results shown in Table 4.3 where the single class U-SVM provides a significant improvement using digit “4” in comparison to digit “3” as universum.

4.7 Conclusion

This chapter introduced single-class U-SVM formulation for *Problem Setting 2*. Setting 2 for single-class learning is (implicitly) adopted in many applications dealing with single-class learning and anomaly detection. The proposed single-class U-SVM can be implemented via minor modification of the binary U-SVM software (http://mloss.org/software/view/19/).

We presented practical conditions for the effectiveness of the single-class U-SVM using histogram of projections. These conditions can be adopted by general users, because:

1. They provide an explicit characterization of the properties of the Universum relative to the properties of training data. These properties are conveniently represented in the form of the univariate histogram of projections;
2. They directly relate prediction performance of single-class U-SVM to that of single-class SVM.

It is important to note that, according to our analyses, meaningful characterization of 'good' Universum is possible only in the context of a particular training dataset

Table 4.1 single-class SVM vs. U-SVM on MNIST (digit "0" as "normal class").

		Single-SVM	**Single U-SVM (digit "1")**	**Single U-SVM (digit "2")**
$\nu = \mathbf{2^{-8}}$ (equivalent $\mathbf{C=2^{-7}}$) ~ Low FN rate				
Training	FN (%)	0.7 (0.2)	0.8(0.2)	0.8 (0.4)
Test	FN (%)	1.8 (0.7)	1.8 (0.6)	1.8 (0.75)
	FP (%) (digits " 3-9")	**69 (3.9)**	**56 (5.2)**	**69 (3.8)**
	FP (%) (on digit " 1")	**4.1 (0.4)**	**0.6 (0.6)**	-
	FP (%) (on digit "2")	**79 (3.1)**	-	**72 (2.7)**
$\nu = \mathbf{2^{-4}}$ **(**equivalent $\mathbf{C=2^{-11.5}}$ **)** ~ High FN rate				
Training	FN (%)	5.9 (0.2)	6.3 (1)	6 (0.3)
Test	FN (%)	6.5 (0.6)	6.5 (0.6)	6.5 (0.6)
	FP (%) (digits " 3-9")	43 (2.4)	43 (3.8)	43 (2.4)
	FP (%) (on digit " 1")	1.5 (0.5)	1.0 (0.7)	-
	FP (%) (on digit "2")	54 (3.3)	-	54 (3.1)

Table 4.2 single-class SVM vs. U-SVM on Reuters-21578 (category "crude" as "normal class").

		Single-SVM	**Single U-SVM ("money-fx")**	**Single U-SVM ("trade")**
$\nu = 2^{-4}$ (equivalent C=2^{-7}) ~ Low FN rate				
Training	FN (%)	11 (3.0)	13 (3.4)	11 (2.5)
Test	FN (%)	28 (0)	29 (2.5)	29 (1.2)
	FP (%) (on *others*)	**14 (1.6)**	**8 (1.1)**	**12 (1.5)**
	FP (%) ("money-fx")	**16 (4.0)**	**0.8 (0.1)**	-
	FP (%) (on "trade")	**29 (1.2)**	**-**	**15 (0.7)**
$\nu = 2^{-2}$ (equivalent C=$2^{-9.5}$) ~ High FN rate				
Training	FN (%)	25 (1.7)	26 (1.5)	26 (2.8)
Test	FN (%)	37 (2.4)	37 (2.4)	37 (2.2)
	FP (%) (on *others*)	12 (2.2)	12 (2.0)	12 (2.0)
	FP (%) ("money-fx")	16 (1.0)	15 (1.9)	-
	FP (%) (on "trade")	35 (3.1)	-	34 (3.2)

Table 4.3 single-class SVM vs. U-SVM on USPS (digit "0" as "normal class").

		Single-SVM	**Single U-SVM (digit "3")**	**Single U-SVM (digit "4")**
2^{-6} (equivalent **$C=2^{-2}$**) ~ Low FN rate				
Training	FN (%)	3.08(0.86)	4.04(0.64)	3.4(1.13)
Test	FN (%)	11.42(0)	11.36(0.12)	11.03(0.24)
	FP (%) (digits " 3-9")	6.55(0)	5.99(0.08)	3.27(0.21)
	FP (%) (on digit " 3")	3.62(0)	0.60(0)	
	FP (%) (on digit "4")	0.7(0.27)		0(0)
$=2^{-3}$ (equivalent **$C=2^{-5.5}$)** ~ High FN rate				
Training	FN (%)	0.23(0.23)	0.87(0.87)	0.61(0.61)
Test	FN (%)	20.3(0)	20.3(0)	20.7(0.16)
	FP (%) (digits " 3-9")	1.4(0)	1.2(0)	0.6(0)
	FP (%) (on digit " 1")	0	0	
	FP (%) (on digit "2")	0		0

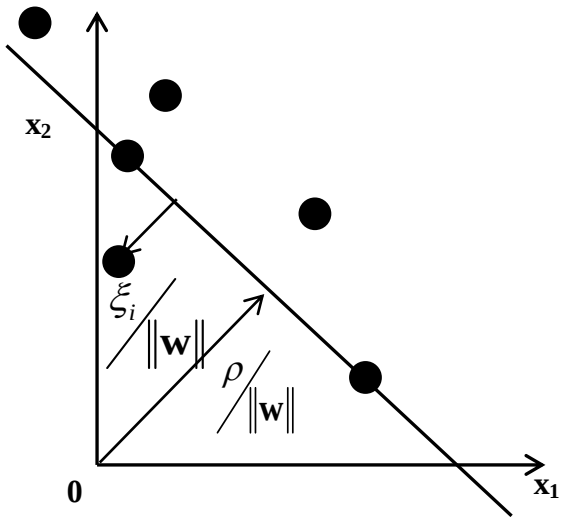
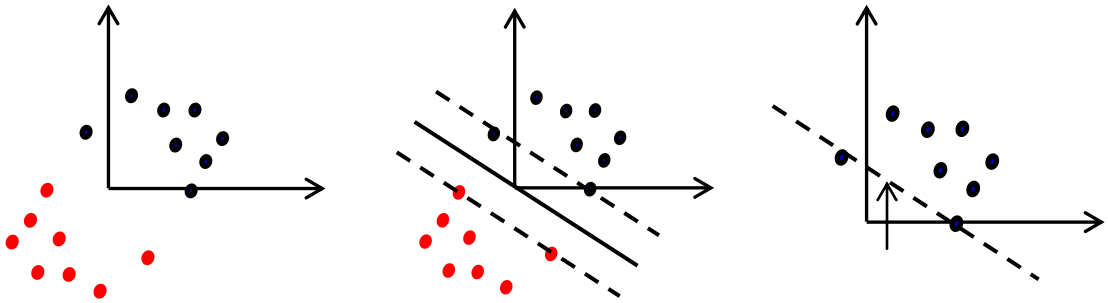
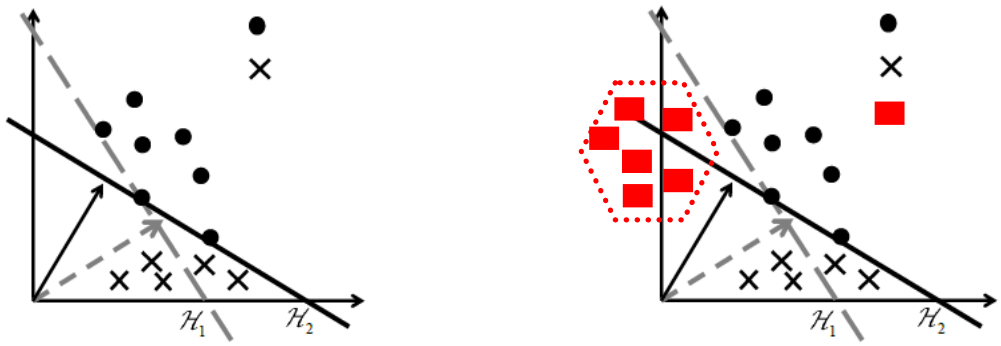


Figure 4.1: Schematic representation of single-class SVM optimization.



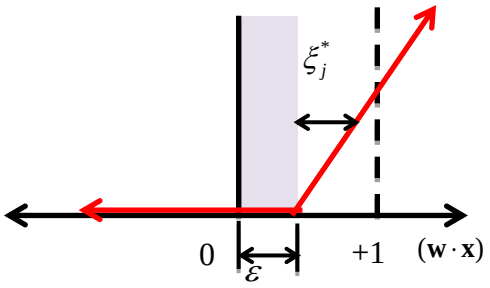
(a) (b) (c)

Figure 4.2: A schematic representation of solving single-class SVM problem using a binary SVM classifier. (a) Reflect training samples about the origin. (b) Estimate a binary SVM classifier. (c) Predict on future test data using the margin as the single-class decision boundary.

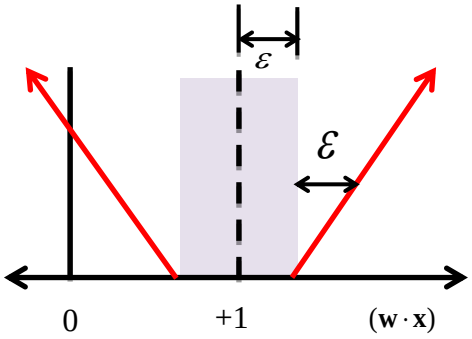


(a) (b)

Figure 4.3: Two large margin hyperplane with different number of contradictions on the Universum. (a) the universum samples *may or may not* follow the same distribution as the "abnormal" class. (b) the universum samples *do not* follow the same distribution as the "abnormal" class.

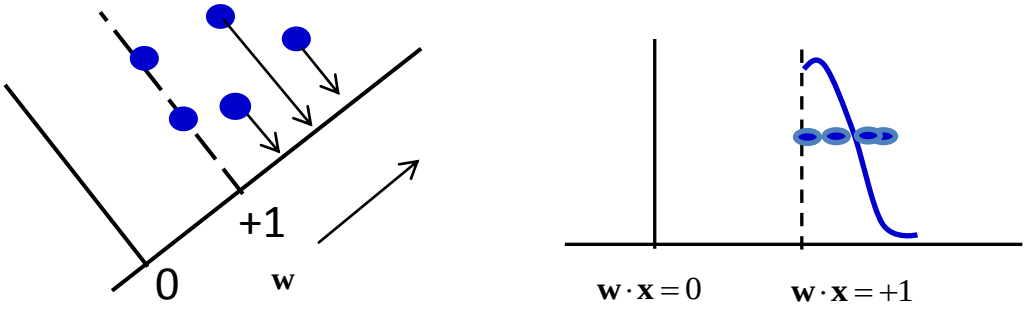


(a)



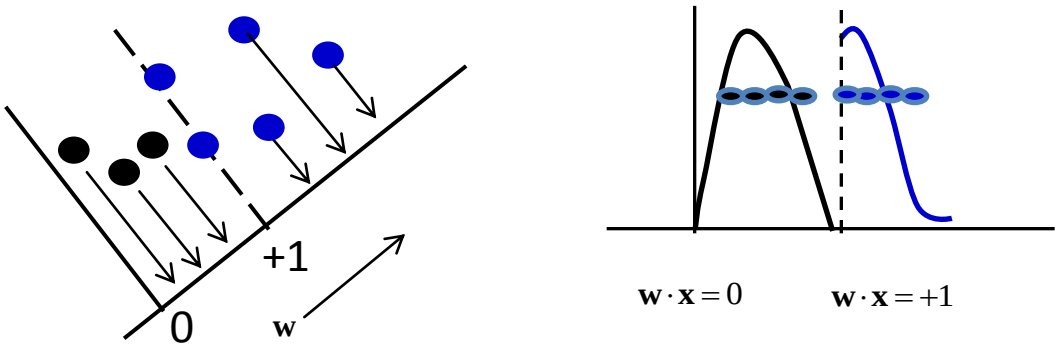
(b)

Figure 4.4: Universum Loss functions. (a)The hinge loss for the Universum samples used in eq. (4.3). (b) The ε - insensitive loss for the Universum samples used in eq. (4.4).



(a) (b)

Figure 4.5: Projection of the training data shown in blue onto the normal weight vector (w) of the one-class SVM hyperplane. (b) Univariate histogram of projections. i.e. histogram of $f(\mathbf{x})$ values for training samples.



(a) (b)

Figure 4.6: Projection of the universum data (shown in black) onto the normal weight vector (w) of the one-class SVM hyperplane. (b) Univariate histogram of projections of the universum samples (shown in black) along with the training samples (shown in blue).

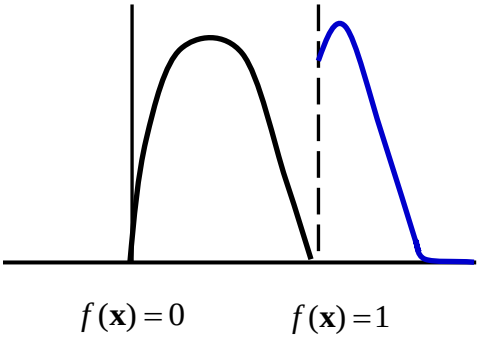
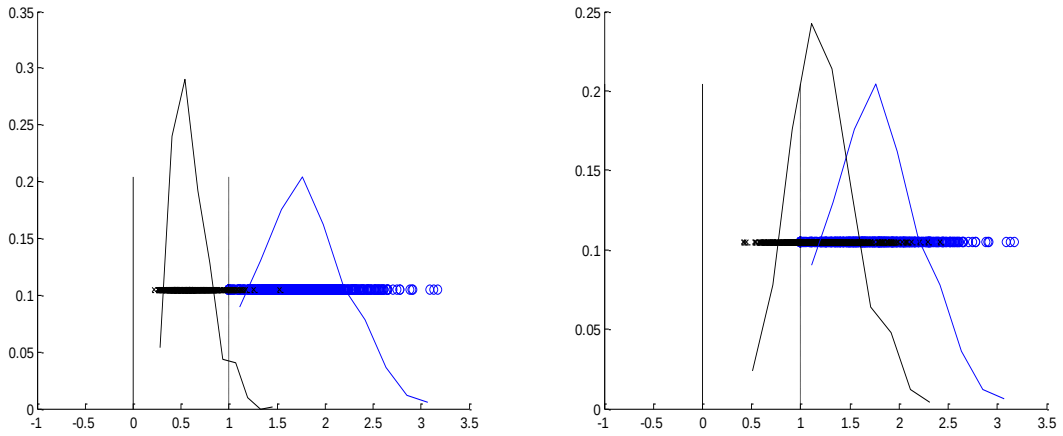
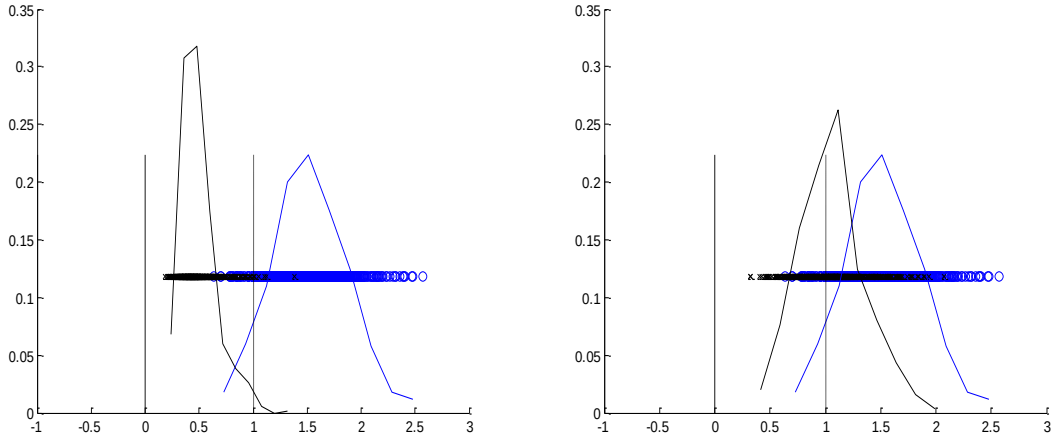


Figure 4.7: A schematic illustration of the histogram of projections of training and universum samples onto normal w vector of SVM decision boundary satisfying the practical conditions for the effectiveness of one-class U-SVM.



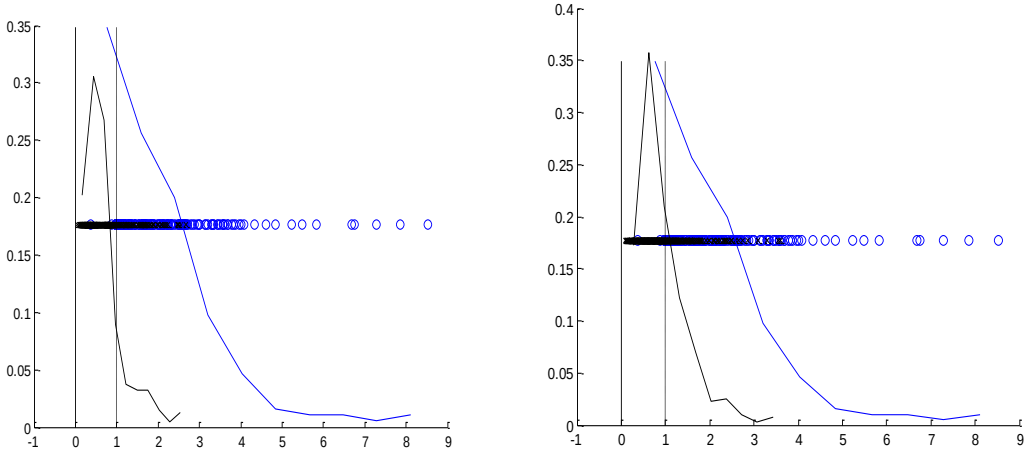
(a) (b)

Figure 4.8: Univariate histogram of projections onto one-class SVM normal weight vector with $C=2^{-7}$ (equivalently $\upsilon = 2^{-8}$) for different types of Universa (for MNIST data). Training set size ~500 samples of digit "0". Universum set size ~500 samples. (a) Digit "1" Universum. $C^*/C=10^{-3}$ (b) Digit "2" Universum. $C^*/C=10^{-3}$.



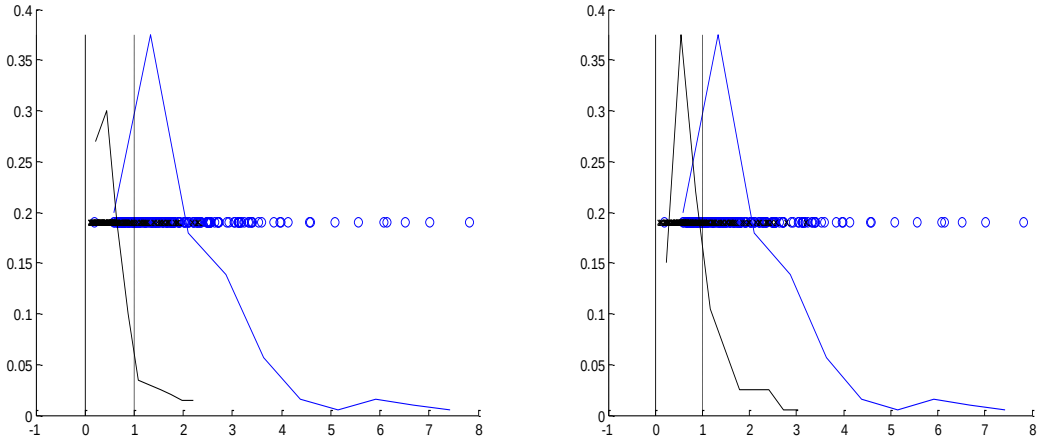
(a) (b)

Figure 4.9: Univariate histogram of projections onto one-class SVM normal weight vector with $C=2^{-11.5}$ (equivalently $\upsilon = 2^{-4}$) for different types of Universa (for MNIST data). Training set size ~500 samples of digit "0". Universum set size ~500 samples. (a) Digit "1" Universum. $C^*/C=10^{-4}$ (b) Digit "2" Universum. $C^*/C=10^{-3}$.



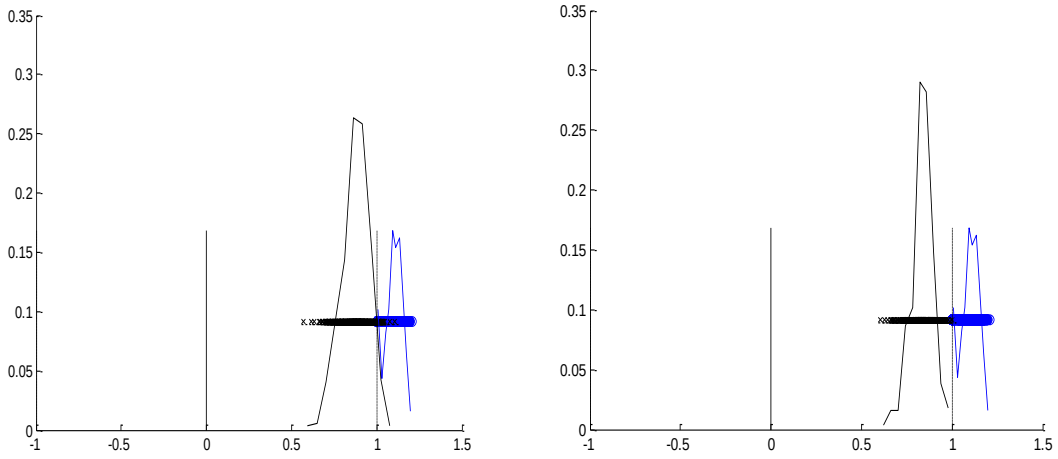
(a) (b)

Figure 4.10: Univariate histogram of projections onto one-class SVM normal weight vector with $C=2^{-2}$ (equivalently $\upsilon = 2^{-8}$) for different types of Universa (for REUTERS-21578 data). Training set size ~195 samples of category "crude". Universum set size ~400 samples. (a) Category "money-fx" Universum. $C^*/C=2$ (b) Category "trade" Universum. $C^*/C=10^{-4}$.



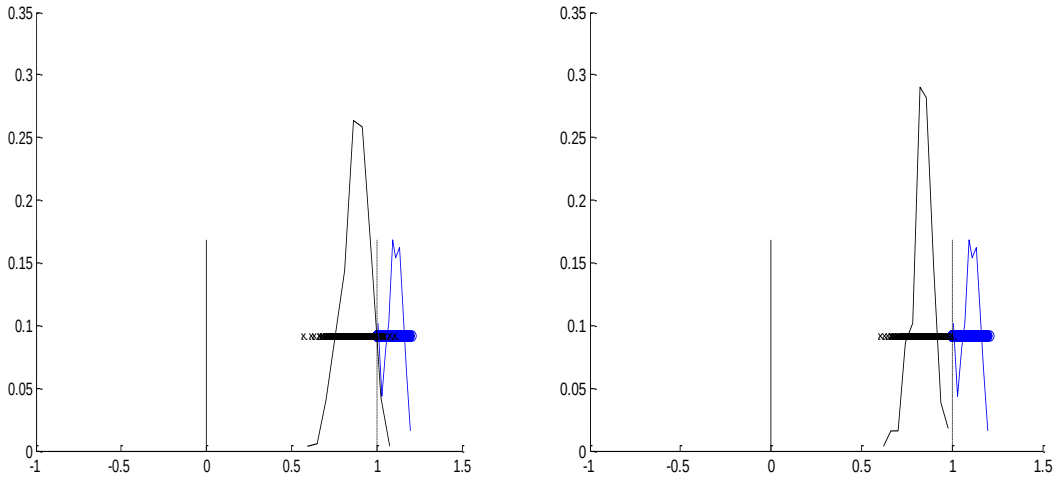
(a) (b)

Figure 4.11: Univariate histogram of projections onto one-class SVM normal weight vector with $C=2^{-7}$ (equivalently $\upsilon = 2^{-4}$) for different types of Universa (for REUTERS-21578 data). Training set size ~195 samples of category "crude". Universum set size ~400 samples. (a) Category "money-fx" Universum. $C^*/C=2^{-1}$ (b) Category "trade" Universum. $C^*/C=10^{-4}$ (c) *noise* Universum $C^*/C=4$.



(a) (b)

Figure 4.12: Univariate histogram of projections onto single-class SVM normal weight vector with $C=2^{-2}$ (equivalently $\upsilon = 2^{-6}$) for different types of Universa (for USPS data). Training set size ~500 samples of digit "0". Universum set size ~500 samples. (a) digit "3" Universum. $C^*/C=2^{-5}$ (b) digit "4" Universum. $C^*/C=2^{-5}$.



(a) (b)

Figure 4.13: Univariate histogram of projections onto single-class SVM normal weight vector with $C=2^{-5.5}$ (equivalently $\upsilon = 2^{-3}$) for different types of Universa (for USPS data). Training set size ~500 samples of digit "0". Universum set size ~500 samples. (a) digit "3" Universum. $C^*/C=2^{-8}$ (b) digit "4" Universum. $C^*/C=2^{-7}$.

Chapter 5 Universum Learning for Regression

5.1 Introduction

The idea of Universum learning or learning through contradiction (Vapnik, 1998; 2006) provides a formal mechanism for incorporating a priori knowledge about the application domain, in the form of additional (unlabeled) Universum samples. However, the implementation of Universum learning is *known only for classification setting*. This chapter extends the notion of Universum learning to *regression* problems. In regression problems the output of the system is a variable that takes on real values: $y \in R$. The task is to estimate real-valued functions, $f(\mathbf{x}, \omega), \omega \in \Omega$ which minimizes the commonly used squared error loss:

$$L(f(\mathbf{x}, \omega), y) = (y - f(\mathbf{x}, \omega))^2 \tag{5.1}$$

For such problems, one can expect to achieve improved generalization performance by incorporating a priori knowledge in the form of additional data samples from the same application domain.

This chapter is organized as follows. Section 5.2 describes standard SVM regression formulation. Section 5.3 extends the notion of Universum learning for regression problems and introduces the new Universum-SVM regression formulation. Next we provide empirical results to illustrate the effectiveness of this new formulation in Section 5.4. Finally, conclusions are presented in Section 5.5.

5.2 SVM Regression

To generalize the SVM algorithm to the regression case, an analog to the margin-based loss function is constructed in the space of the target values $y \in R$ by using Vapnik's (1998, 2006) ε-insensitive loss $L_\varepsilon(f(\mathbf{x},\mathbf{w}),y) = \max(|y - f(\mathbf{x},\mathbf{w})| - \varepsilon, 0)$ (see Figure 5.1). The ε-insensitive loss can be formally described by introducing non-negative slack variables $\xi_i, \xi_i^*, i = 1,\ldots,n$ to measure the deviation of training samples outside ε-insensitive zone (Figure 5.2). The empirical risk can then be expressed as sum of the slack variables: $R_{\text{emp}}(\mathbf{w}) = \sum_{i=1}^{n}(\xi_i + \xi_i^*)$. SVM regression attempts to strike a balance between minimization of empirical risk and the penalization term:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}(\mathbf{w}\bullet\mathbf{w}) + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{subject to:} \quad & y_i - (\mathbf{w}\bullet\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ & (\mathbf{w}\bullet\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \ge 0,\ i = 1,\ldots,n. \end{aligned} \tag{5.2}$$

The parameter C controls the trade-off between the empirical risk and the penalization term. In practice, an optimal value of C can be determined based on the range of response (y) values (Cherkassky and Ma, 2004). Parameter ε controls the model complexity (size of margin) and is tuned through model selection.

Problem (5.2) is usually solved in its dual form:

$$\begin{aligned} \min_{\alpha,\beta} \quad & \varepsilon\sum_{i=1}^{n}(\alpha_i + \beta_i) - \sum_{i=1}^{n} y_i(\alpha_i - \beta_i) + \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \beta_i)(\alpha_j - \beta_j)(\mathbf{x}_i\bullet\mathbf{x}_j) \\ \text{subject to:} \quad & \sum_{i=1}^{n}\alpha_i = \sum_{i=1}^{n}\beta_i \\ & 0 \le \alpha_i \le C,\ 0 \le \beta_i \le C,\ i = 1,\ldots,n. \end{aligned} \tag{5.3}$$

The values of parameters $\alpha_i^*, \beta_i^*, i = 1, \ldots, n$ found by solving problem (5.3) give the SVM regression function:

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i^* - \beta_i^*)(\mathbf{x}_i \bullet \mathbf{x}) + b \tag{5.4}$$

The samples with non-zero coefficients are support vectors (SVs), corresponding to data points at or outside ε-insensitive zone. The bias term b is given by

$$b = y_s - \sum_{i=1}^{n} (\alpha_i^* - \beta_i^*)(\mathbf{x}_i \bullet \mathbf{x}_s)$$

where $(\mathbf{x}_s, y_s)$ is a support vector.

The linear SVM can be extended for nonlinear cases by mapping input vectors $\mathbf{x}$ into a high dimensional feature space Z through some nonlinear mapping $\mathbf{z} = \phi(\mathbf{x})$, and constructing the optimal hyperplane in the feature space (Vapnik, 1998). Note that we do not need to specify $\mathbf{z} = \phi(\mathbf{x})$ explicitly. The only thing we need to do is to replace the dot product $(\mathbf{x}_i \bullet \mathbf{x}_j)$ in (5.3) with some kernel $K(\mathbf{x}_i, \mathbf{x}_j)$. Commonly used kernel functions include:

- Polynomial kernel (of degree q)

$$K(\mathbf{x}_i, \mathbf{x}_j) = ((\mathbf{x}_i \bullet \mathbf{x}_j) + 1)^q$$

- Radial Basis Function (RBF) kernel (with width parameter γ)

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \| \mathbf{x}_i - \mathbf{x}_j \|^2)$$

5.3 Universum-SVM Regression

In this chapter, we develop a new Universum-SVM regression formulation, based on *formalizing the notion of falsification* for regression, as explained next. Consider a regression setting where available training data $\{(\mathbf{x_i}, y_i), i=1,2,.., n\}$ is modeled by linear SVM regression. For SVM regression, the concept of 'margin' is implemented via ε - insensitive zone (see Fig. 5.1). That is, training samples falling inside this zone are 'explained' by SVM model, and samples outside 'falsify' the model (Cherkassky and Mulier 2007). Next, consider two SVM regression models which explain training samples *equally well*. That is, both SVM models use the same value of ε and achieve the same value of ε - insensitive loss for training samples (e.g., zero loss as shown in Fig. 5.3). Now, consider additional *Universum* samples $\{(\mathbf{x_j^*}, y_j^*), j=1, 2,\ldots, m\}$. These samples are defined in the same $(\mathbf{x}, y)$ space as the training samples, and they reflect a priori knowledge that they cannot be explained by a regression model, or equivalently, they *should be falsified* by a regression model. Hence, we should favor the model for which universum samples *lie outside* the ε - insensitive zone. As shown in Fig. 5.3, the SVM model shown in black should be favored. Note that the Universum samples (for regression) are *labeled*, unlike unlabeled Universum samples in the original U-SVM classifier. This reasoning motivates new Universum SVM (U-SVM) regression formulation. Standard ε-insensitive loss is used for training samples and it forces them to lie inside ε - insensitive tube. However, the Universum samples are penalized by a different loss function shown in Fig. 5.4(a). This loss function forces the universum samples to lie outside a $\pm\Delta$ zone. Penalization of the universum samples inside this $\pm\Delta$

zone is technically achieved via the slack variables ζ_j as shown in Fig. 5.4(a). Note that the tunable parameter Δ can be larger (or smaller) than ε. This leads to new formulation for U-SVM regression:

(Universum Loss $U_\Delta(y_j^* - f(\mathbf{x}_j^*))$

$$\min_{\mathbf{w},b,\xi_i,\xi_i',\zeta_j} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \quad + \quad C^*\sum_{j=1}^{m}\zeta_j \qquad (5.5)$$

subject to,

$$y_i - (\mathbf{w}\cdot\mathbf{x}_i + b) \le \varepsilon + \xi_i$$
$$(\mathbf{w}\cdot\mathbf{x}_i + b) - y_i \le \varepsilon + \xi_i'$$
$$\xi_i, \xi_i' \ge 0 \text{ and } i = 1 \text{ to } n$$
(convex)

$$\left|y_j^* - (\mathbf{w}\cdot\mathbf{x}_j^* + b)\right| \ge \Delta - \zeta_j$$
$$\zeta_j \ge 0 \text{ and } j = 1 \text{ to m}$$
(concave)

Here parameters $\varepsilon, \Delta \ge 0$ and $C, C^* \ge 0$ control the tradeoff between explaining the training samples and 'falsification' of the universum samples.

Note that formulation (5.5) is non-convex due to the Universum loss $U_\Delta(y_j^* - f(\mathbf{x}_j^*))$ (see fig. 5.4 a). Recently, similar non-convex optimization problems have been addressed in (Shen 2003, Collobert 2006) using the ConCave Convex Programming (CCCP) strategy. For the CCCP strategy we decompose the cost function J(θ) as the sum of a convex part $J_{vex}(\theta)$ and a concave part $J_{cav}(\theta)$, where θ is the optimization argument. Each iteration of the CCCP procedure approximates the concave part by its tangent and minimizes the resulting convex function (see Algorithm 5.1).

Algorithm 5.1 : The ConCave Convex Programming (CCCP)

Initialize θ^0 with a best guess

Repeat

$$\theta^{t+1} = \arg\min_{\theta} \; (\; J_{vex}(\theta) + \; J_{cav}'(\theta^t)\cdot\theta)$$

Until convergence of θ^t

In this chapter we use the CCCP strategy to solve the non-convex optimization formulation (5.4). The main contribution of this section is the derivation of the CCCP updates to solve the U-SVM regression formulation in (5.5). This leads to our proposed Algorithms 5.2 and 5.3 to solve the U-SVM regression problem in its primal and dual forms respectively.

The Universum loss function can be represented as a sum of two ramp losses, $U_\Delta(y_j^* - f(\mathbf{x}_j^*)) = \quad A_\Delta(y_j^* - f(\mathbf{x}_j^*)) + \quad A_\Delta(-y_j^* + f(\mathbf{x}_j^*)) + \quad$ a constant ; where $A_\Delta(t) = \underbrace{\max(0, \Delta - t)}_{convex} \quad \underbrace{-\max(0, -t)}_{concave}$ (see Fig. 5.4 b). The constant term does not affect the optimization; and hence (5.4) can be re-written as,

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,\boldsymbol{\zeta},\boldsymbol{\zeta}^*} \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + C^*\sum_{j=1}^{m}(\zeta_j + \zeta_j^*)}_{convex} - \underbrace{C^*\sum_{j=1}^{2m} H(y_j^*, f(\mathbf{x}_j^*))}_{concave} \quad (5.6)$$

subject to :

$$y_i - f(\mathbf{x}_i) \le \varepsilon + \xi_i \;, \qquad \xi_i \ge 0 \qquad i = 1 \text{ to } n$$

$$f(\mathbf{x}_i) - y_i \le \varepsilon + \xi_i^* \qquad \xi_i^* \ge 0 \qquad i = 1 \text{ to } n$$

$$y_j^* - f(\mathbf{x}_j^*) \le -\Delta + \zeta_j \qquad \zeta_j \ge 0 \qquad j = 1 \text{ to m}$$

$$f(\mathbf{x}_j^*) - y_j \le -\Delta + \zeta_j^* \qquad \zeta_j^* \ge 0 \qquad j = 1 \text{ to m}$$

where, $H(y_j^*, f(\mathbf{x}_j^*)) = \begin{cases} \max\ (0, -y_j^* + f(\mathbf{x}_j^*)) ; & \text{j=1 to m} \\ \max\ (0,\ \ y_j^* - f(\mathbf{x}_j^*)) ; & \text{j=m+1 to 2m} \end{cases}$

and $f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b$.

Define,

$$k_j = -C^* \frac{\partial H(y_j^* - f(\mathbf{x}_j^*))}{\partial f(\mathbf{x}_j^*)} = \begin{cases} -C^* & \text{if } y_j^* < f(\mathbf{x}_j^*) \text{ and j=1 to m} \\ C^* & \text{if } y_j^* > f(\mathbf{x}_j^*) \text{ and j=m+1 to 2m} \\ 0 & \text{else} \end{cases} \quad (5.7)$$

Then following the CCCP strategy, we have the following Algorithm 5.2.

Algorithm 5.2: CCCP algorithm for U-SVM regression.

---

1. Initialize $(\mathbf{w}^0, b^0)$ using the standard SVM regression model.

Repeat,

2. At t+1 iteration update, $k_j^{t+1} = \begin{cases} -C^* & \text{if } y_j^* < (\mathbf{w}^t \cdot \mathbf{x}_j^*) + b \text{ and j=1 to m} \\ C^* & \text{if } y_j^* > (\mathbf{w}^t \cdot \mathbf{x}_j^*) + b \text{ and j=m+1 to 2m} \\ 0 & \text{else} \end{cases}$

3. Solve the following eq. (5.6) to obtain $(\mathbf{w}^{t+1}, b^{t+1})$

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,\boldsymbol{\zeta},\boldsymbol{\zeta}^*} \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) + C^*\sum_{j=1}^{2m}(\zeta_j + \zeta_j^*)}_{convex} - \underbrace{\sum_{j=1}^{2m} k_j^t[(\mathbf{w}\cdot\mathbf{x}_j^*) + b]}_{concave} \quad (5.8)$$

$$\begin{aligned} \text{subject to: } & y_i - f_t(\mathbf{x}_i) \le \varepsilon + \xi_i, & \xi_i \ge 0 \quad & i = 1 \text{ to } n \\ & f_t(\mathbf{x}_i) - y_i \le \varepsilon + \xi_i^* & \xi_i^* \ge 0 \quad & i = 1 \text{ to } n \\ & y_j^* - f_t(\mathbf{x}_j^*) \le -\Delta + \zeta_j & \zeta_j \ge 0 \quad & j = 1 \text{ to m} \\ & f_t(\mathbf{x}_j^*) - y_j \le -\Delta + \zeta_j^* & \zeta_j^* \ge 0 \quad & j = 1 \text{ to 2m} \end{aligned}$$

4. **Until** convergence i.e. $k_j^{t+1} = k_j^t \quad \forall\, j = 1 \text{ to } 2m$

The Algorithm (5.2) can be extended to nonlinear case by solving the problem in its dual form (as shown next),

Rewrite, $\mathbf{x}_i = \begin{cases} \mathbf{x}_i & \text{i=1 to n} \quad \text{(training samples)} \\ \mathbf{x}_j^* & \text{j=1 to m ; i= n+1 to n+m} \quad \text{(universum samples)} \end{cases}$

$$\rho_i = \begin{cases} \varepsilon & \text{i=1 to n} \\ -\Delta & \text{i=n+1 to n+m} \end{cases} \quad \text{and} \quad C_i = \begin{cases} \text{C} & \text{i=1 to n} \\ \text{C*} & \text{i=n+1 to n+m} \end{cases}$$

$$\delta_i = \begin{cases} C^* & \text{if } y_i < f(\mathbf{x}_i) \quad \text{and i=n+1 to n+m} \\ 0 & \text{else} \end{cases}$$

$$\gamma_i = \begin{cases} C^* & \text{if } y_i > f(\mathbf{x}_i) \quad \text{and i=n+1 to n+m} \\ 0 & \text{else} \end{cases}$$

Then, in the dual form we use the following Algorithm (5.3). The proof follows from the standard KKT equations and has been omitted from this chapter.

Algorithm 5.3: CCCP algorithm for U-SVM regression in dual form.

---

1. Initialize $[\alpha^0, \beta^0, b^0]$ using the standard SVM regression model (see eq. 5.3)

Repeat,

2. At t+1 iteration update,

$$\delta_i^{t+1} = \begin{cases} C^* & \text{if } y_i < \sum_{i=1}^{n+m} (\alpha_i^t - \beta_i^t)(\mathbf{x}_i \cdot \mathbf{x}) + b \quad \text{and i=n+1 to n+m} \\ 0 & \text{else} \end{cases}$$

$$\gamma_i^{t+1} = \begin{cases} C^* & \text{if } y_i > \sum_{i=1}^{n+m} (\alpha_i^t - \beta_i^t)(\mathbf{x}_i \bullet \mathbf{x}) + b \quad \text{and i=n+1 to n+m} \\ 0 & \text{else} \end{cases}$$

3. Solve the following eq. (5.7) to obtain $(\alpha^{t+1}, \beta^{t+1}, b^{t+1})$

$$\min_{\alpha,\beta} \quad \sum_{i=1}^{n+m} \rho_i(\alpha_i + \beta_i) - \sum_{i=1}^{n+m} y_i(\alpha_i - \beta_i) + \frac{1}{2} \sum_{i,j=1}^{n+m} (\alpha_i - \beta_i)(\alpha_j - \beta_j) K(\mathbf{x}_i \bullet \mathbf{x}_j) \qquad (5.9)$$

$$\text{subject to: } \sum_{i=1}^{n+m} \alpha_i = \sum_{i=1}^{n+m} \beta_i, \quad -\gamma_i^t \le \alpha_i \le C_i - \gamma_i^t, \quad -\delta_i^t \le \beta_i \le C_i - \delta_i^t, \quad i = 1 \text{ to } n+m$$

4. **Until** $\gamma_i^{t+1} = \gamma_i^t$ and $\delta_i^{t+1} = \delta_i^t \quad \forall$ i = 1 to n+m

---

This CCCP based non-convex minimization could get stuck in local optima, and hence a good initialization and stopping criteria are crucial for this algorithm. In our implementation, standard SVM regression model is used as the initial condition (see Algorithm 5.2 and 5.3). Thus the CCCP strategy searches for local minimum near (standard) SVM regression solution. Further, at each iteration we are solving an SVM-like formulation. The only modification required is to the constraints as shown in eq. (5.9). Hence, the time complexity to solve the U-SVM regression using CCCP involves solving the standard SVM regression formulation for (n+m) samples at each iteration. Our preliminary experience suggests fast convergence (5-10 iterations) for several data sets. Hence, this strategy is scalable for most real-life datasets.

The kernelized version of U-SVM regression formulation (5.4) has five tunable parameters: C, C*, kernel parameter, ε and Δ. So model selection (parameter tuning) becomes an issue for any real-life application. We propose the following model selection strategy for estimating a U-SVM regression model:

1. Fix C = y_{max} - y_{min} (following Cherkassky and Mulier, 2007) and perform model selection for ε and kernel parameters.

2. perform model selection for the C*/C and Δ parameter specific to the U-SVM regression formulation, while keeping C, ε and kernel parameters fixed. This can be performed using resampling or a separate validation set (as in empirical results presented in Section 5.3).

5.4 Empirical Results for U-SVM Regression

This section presents initial empirical results to show the effectiveness of the proposed U-SVM regression formulation relative to standard SVM regression.

Our *experiment* uses a synthetic 30-dimensional hypercube data set, where each input is uniformly distributed in [-1, 1] interval. The output is generated as $y = (x_1+\ldots+x_5- x_6-\ldots- x_{10} + x_{11}+\ldots+x_{15}- x_{16}-\ldots-x_{20}+x_{21}+\ldots+x_5- x_{26}-\ldots-x_{30}+\xi)$; where the noise follows a normal distribution : $\xi \sim N(0,\sigma)$. For this data set, only linear SVM is used because the optimal model is known to be linear. Here, we use two different types of universum,

Universum 1: input is same as the training samples $\mathbf{x} \in \mathbf{R}^{30}$ and is uniformly distributed in [-1, 1]. However the output is generated as $y = (-x_1-\ldots-x_5 + x_6+\ldots+x_{10} - x_{11}-\ldots-x_{15}+ x_{16}+\ldots+x_{20}-x_{21}-\ldots-x_5+x_{26}+\ldots+x_{30}+\xi)$.

Universum 2: input is same as the training samples $\mathbf{x} \in \mathbf{R}^{30}$ and is uniformly distributed in [-1, 1]. However the output y ~ uniformly distributed noise within the range of the y values of the training samples.

For this experiment we use the following settings,

- No. of training/validation samples = 30, 90 (characterizing *low, high* sample size settings respectively. An independent validation data set is used for model selection. The number of validation samples is set to be the same as the number of training samples).
- No. of test samples = 5000.
- No. of universum samples = 120.

Further, for the low sample size case we use two different noise levels σ = 0.5, 2 to

capture the effects of low and high noise levels respectively. Similarly, for the high sample size case we use the noise levels σ = 0.5, 2, 4 to capture the effects of low, moderate and high noise levels respectively.

Next we provide the modeling results for standard SVM and U-SVM regression using linear kernel. For our empirical comparisons, we provide the *normalized root mean squared error* ($NRMS = \frac{\sqrt{MSE}}{std(y)}$). Note that, following (Drucker et al, 1997) we drop the noise while reporting the NRMS values. For example, suppose for a given function $y = f(\mathbf{x}) + noise;\ \mathbf{x} \in R^d$ we make a prediction $\hat{y}$ using our estimated model; here d is the dimension of the input space. Then, we report $NRMS = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - \hat{y}_i)^2}}{std(f(\mathbf{x}))}$; where n = no. of test samples. The model selection is performed by tuning parameter values providing the smallest NRMS on the independent validation set. Table 5.1 and 5.2 shows performance comparisons for the standard SVM, and U-SVM. The tables show the average NRMS values on the test samples over 10 random experiments. Here, for each experiment we randomly select the training/validation/test set. The standard deviation of the NRMS values is shown in parenthesis.

For a detailed analysis of the results we adopt the technique of the "histogram of projections" originally introduced for classification. Under classification setting, the 'projection value' for a given sample measures its distance from SVM decision boundary. For regression, conceptually similar quantity is the residual $y - f(\mathbf{x})$ that measures the difference between response *y* and its estimate $f(\mathbf{x})$. So for regression models we use the

univariate histogram of residual values $y - f(\mathbf{x})$ of the training samples, where $f(\mathbf{x})$ is the trained regression model with optimally tuned parameters (see Fig.5.5). In addition to that we also project the *residual values* ($y^* - f(\mathbf{x}^*)$) of the universum samples (shown in black). The estimated $\pm\varepsilon$ value is shown in black dashed lines and the $\pm\Delta$ value is shown in green dashed line. Note that the $\pm\Delta$ parameter is specific only to the U-SVM model. The $\pm\Delta$ lines in the histograms of the SVM models are used for the purpose of analysis of the results.

Low sample size settings (no. of training samples =30)

- **Low Noise (σ = 0.5)**

  As seen from the histograms in Fig. 5.6 and Fig. 5.8 the residual values for the training samples are heavily piled onto the $\pm\varepsilon$ value. This is better illustrated in the Figs. 5.7 and 5.9, where we scale the histograms to the range of [-2, 2]. Typically, for high-dimensional sparse data sets most training samples tend to cluster near the $\pm\varepsilon$ boundaries due to data piling effect similar to classification. See Fig. 5.7a and Fig. 5.9a. Surprisingly, this effect is not well-known for SVM regression problems. However, this phenomenon has been noted in empirical comparisons of various loss functions for regression (Boyd et al 2004, page 296).

  Further,

  Univerum 1: Application of U-SVM provides better generalization than the SVM model (see Table 5.1). Here, U-SVM model (Fig. 5.6b) results in sparser distributed of universum samples within the $\pm\Delta$ zone (shown in green) in

comparison to the SVM model (see Fig. 5.6a). Here, the U-SVM model well-explains the training samples (which lies within the $\pm\varepsilon$ zone) and increases the contradiction on the universum samples (forcing them to lie outside the $\pm\Delta$ zone).

Univerum 2: Same as above, here the U-SVM model (Fig. 5.8b) results in sparser distributed of universum samples within the $\pm\Delta$ zone (shown in green) in comparison to the SVM model (see Fig. 5.8a). Application of the U-SVM improves the prediction accuracy by resulting into an even sparser distributed of the universum within the $\pm\Delta$ zone in comparison to the SVM model.

- **High noise (σ = 2)**

For this case, the data piling effect is not as prominent as before. This shows that higher noise levels results in lower data piling effects.

Further,

Univerum 1: U-SVM regression model (Fig. 5.10b) provides slight improvement over the standard SVM model (see Table 5.1), and results in sparser distributed of the universum samples near $\pm\Delta$ zone in comparison to the SVM model (in Fig. 5.10a).

Univerum 2: Application of the U-SVM does not provide any notable change to the initial SVM solution and hence provides no improvement over the standard SVM model (see Fig. 5.11)

High sample size settings (no. of training samples = 90)

- **Low noise (** σ = 0.5)

  For this case, we do not have data piling onto the $\pm\varepsilon$ values. Comparison of the results in Table 5.2 shows that,

  Univerum 1: Application of the U-SVM does not provide any notable change to the initial SVM solution and hence provides no improvement over the standard SVM model (also seen in Fig. 5.12)

  Univerum 2: Application of the U-SVM does not provide any notable change to the initial SVM solution and hence provides no improvement over the standard SVM model (also seen in Fig. 5.13)

- **Moderate noise (** σ = 2)

  For this case, we do not observe data piling effect about the $\pm\varepsilon$ value. Further,

  Univerum 1: U-SVM regression model provides no improvement over the standard SVM model (see Table 5.2 and Fig. 5.14).

  Univerum 2: In this case application of the U-SVM regression model provides no improvement over the SVM model (see Table 5.2 and Fig. 5.15).

- **High noise (** σ = 4)

  For this case, we do observe some data piling effect about the $\pm\varepsilon$ value. Further,

  Univerum 1: U-SVM regression model provides no significant improvement over the standard SVM model (see Table 5.2 and Fig. 5.16).

  Univerum 2: In this case application of the U-SVM regression model provides no improvement over the SVM model (see Table 5.2 and Fig. 5.17).

Our results indicate that U-SVM regression is effective under high data piling conditions. Such data piling effect is typical in sparse high-dimensional data sets settings. Under such sparse high-dimensional data settings U-SVM regression could provide better generalization than standard SVM regression.

5.5 Conclusion

This chapter introduced U-SVM for regression problems. The proposed U-SVM regression formulation is non convex and can be solved using the ConCave Convex Programming (CCCP) strategy. Further, we provide a sound practical strategy for tuning model parameters in the proposed U-SVM regression. Finally, we provide empirical results to show the effectiveness of the proposed U-SVM regression over standard SVM regression. Our analysis suggests that U-SVM regression is particularly effective for sparse high-dimensional data settings where the histogram of projections for the residual values show data piling about the $\pm\varepsilon$ value.

However, the effectiveness of U-SVM regression also depends on the property of the universum samples. There is still a need to provide better characterization of the 'good' Universa, for which U-SVM can provide improvement over the SVM regression. Deriving the practical conditions for the effectiveness of U-SVM regression has not been covered in this chapter and is left for future research.

Table 5.1 Comparison of average test error (NRMS) for different Universa for low sample size settings (no. of training samples = 30).

NRMS Error in %	SVM	U-SVM (type1)	U-SVM (type2)
Low noise: no. of training samples =30 with $\sigma = 0.5$			
Test	42.86 (8.81)	34.78 (6.83)	37.51 (6.39)
Training	16.18 (2.68)	15.65 (2.43)	16.12 (2.87)
High noise: no. of training samples =30 with $\sigma = 2$			
Test	75.34 (8.89)	73.66 (9.93)	75.29 (6.64)
Training	44.24 (8.03)	44.89 (9.71)	43.68 (6.91)

Table 5.2 Comparison of average test error (NRMS) for different Universa for high sample size settings (no. of training samples = 90).

NRMS Error in %	SVM	U-SVM (type1)	U-SVM (type2)
Low noise: no. of training samples =90 with $\sigma = 0.5$			
Test	11.58(1.57)	11.06(0.8)	11.27(1.52)
Training	10.66(1.76)	10.43(1.68)	10.09(1.99)
Moderate noise: no. of training samples =90 with $\sigma = 2$			
Test	52.16(2.3)	50.75(2.98)	51.1(2.28)
Training	36.38(4.86)	35.58(5.14)	36.12(5.06)
High noise: no. of training samples =90 with $\sigma = 4$			
Test	80.17(6.83)	77.43(6.59)	80.07(9.25)
Training	50.63(9.68)	44.15(8.34)	50.59(10.07)

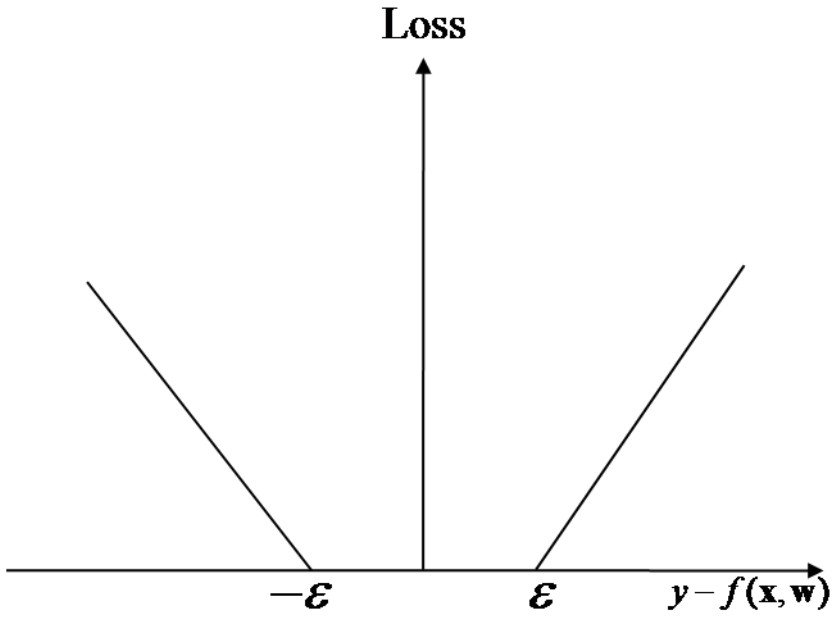


Figure 5.1 ε -insensitive loss function for SVM regression

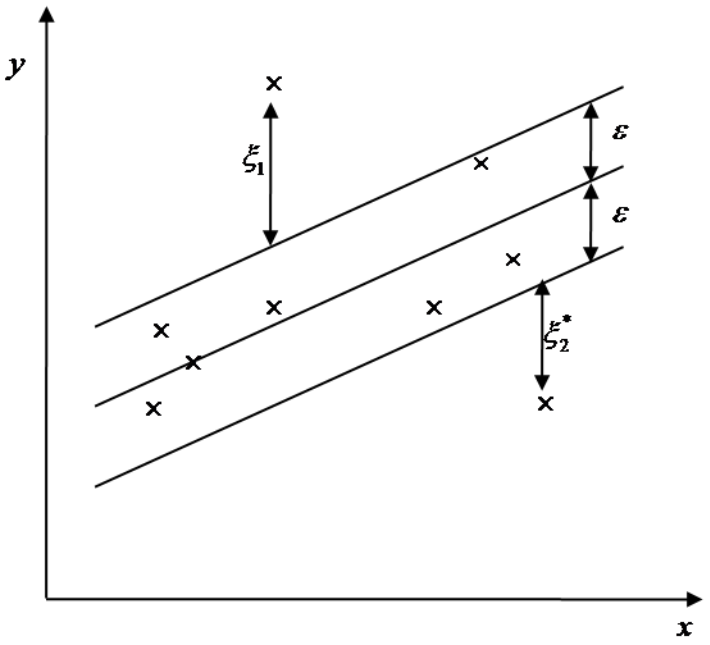


Figure 5.2 slack variable ξ for linear SVM regression formulation.

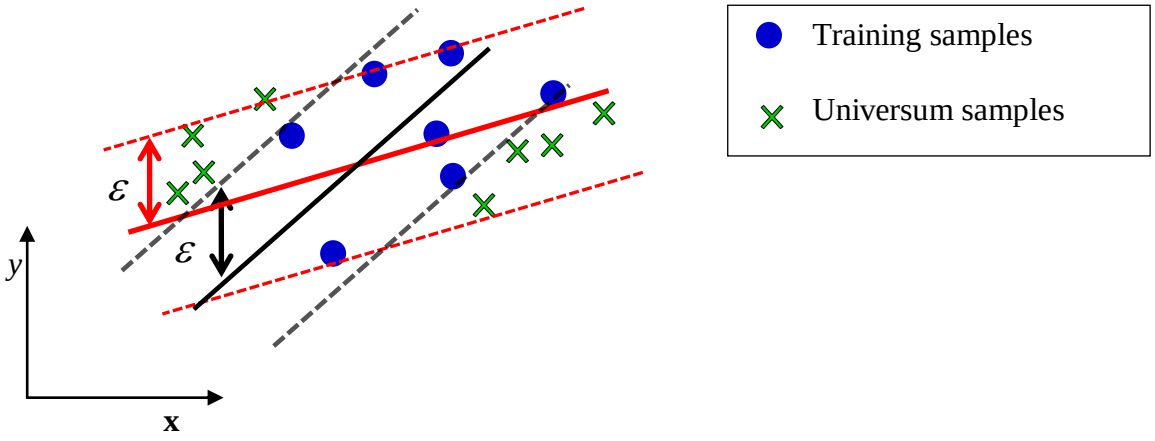


Figure 5.3 Two SVM regression models explain training data equally well, but have different number of contradictions on the Universum. The model with a larger number of contradictions (in black) is selected.

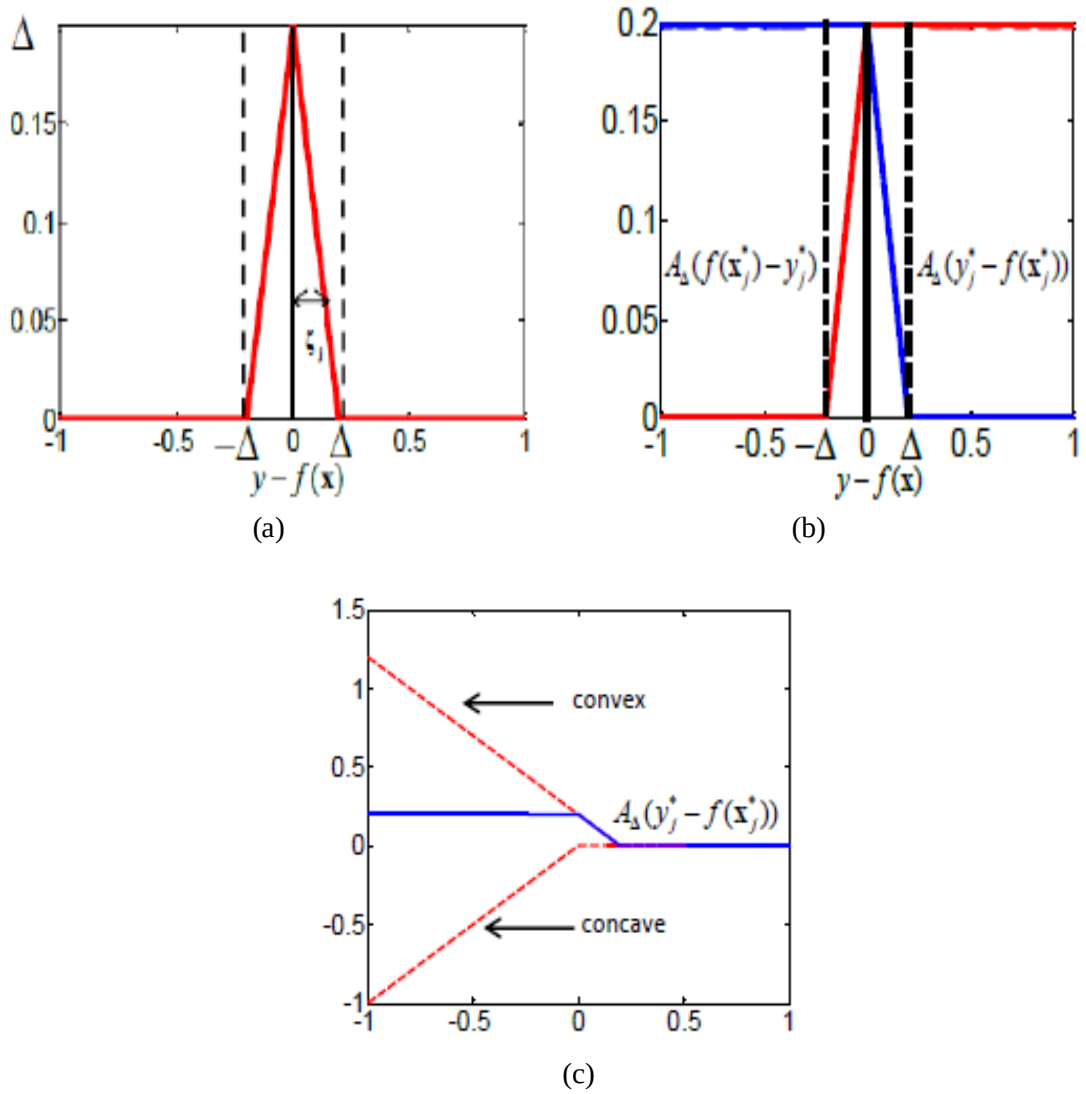


Figure 5.4. (a) Loss function for the universum samples $U_{\Delta}(y_j^* - f(\mathbf{x}_j^*))$ with $\Delta = 0.2$. (b) Universum loss as the sum of two ramp losses $A_{\Delta}(y_j^* - f(\mathbf{x}_j^*))$ and $A_{\Delta}(f(\mathbf{x}_j^*) - y_j^*)$. (c) Decomposition of $A_{\Delta}(y_j^* - f(\mathbf{x}_j^*))$ as the sum of a convex and concave loss.

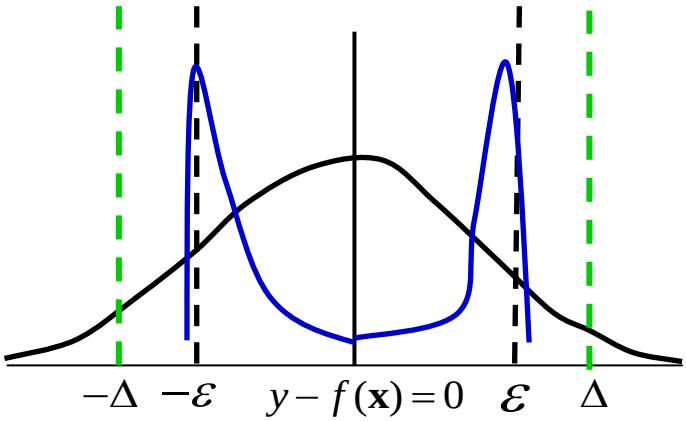
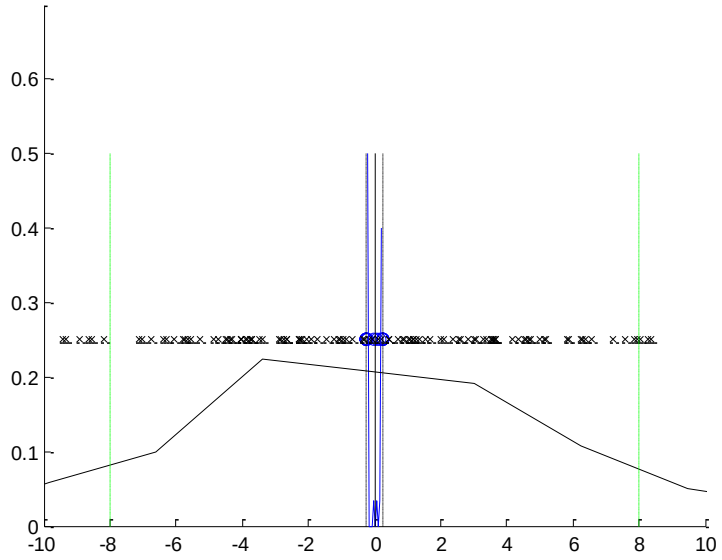
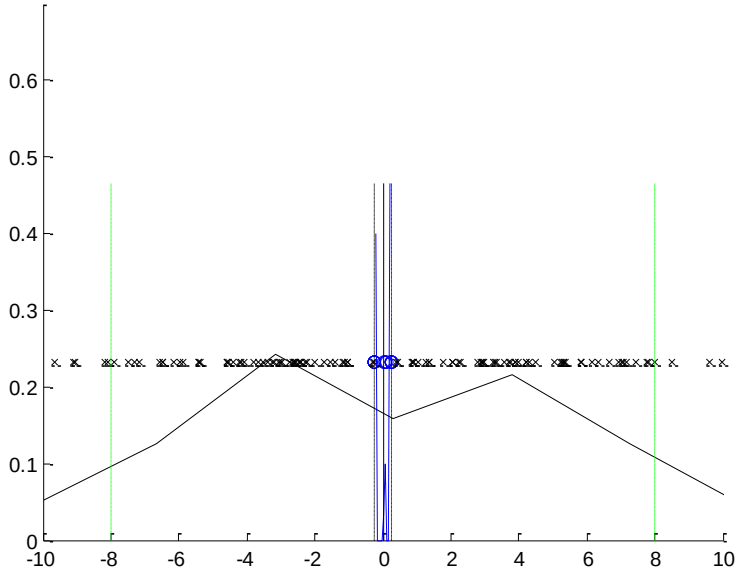


Figure 5.5 Representation of the histogram of residuals $y - f(\mathbf{x})$ for estimated regression model. Training samples are shown in blue and Universum samples shown in black. The estimated $\pm\varepsilon$ value is shown in black dashed lines and the $\pm\Delta$ value is shown in green dashed line.



(a)

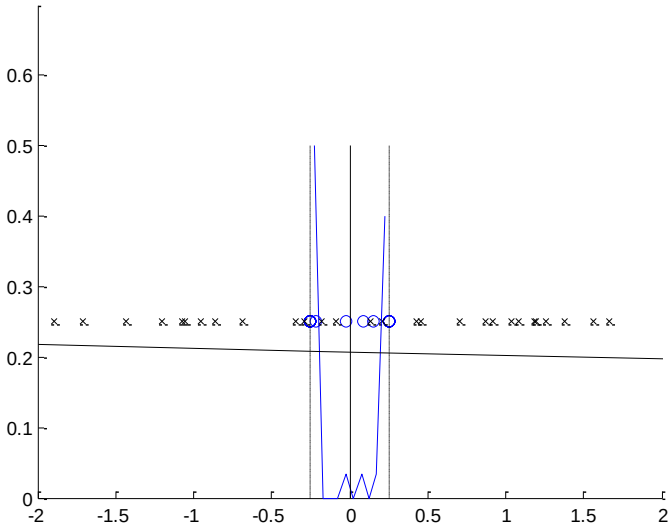


(b)

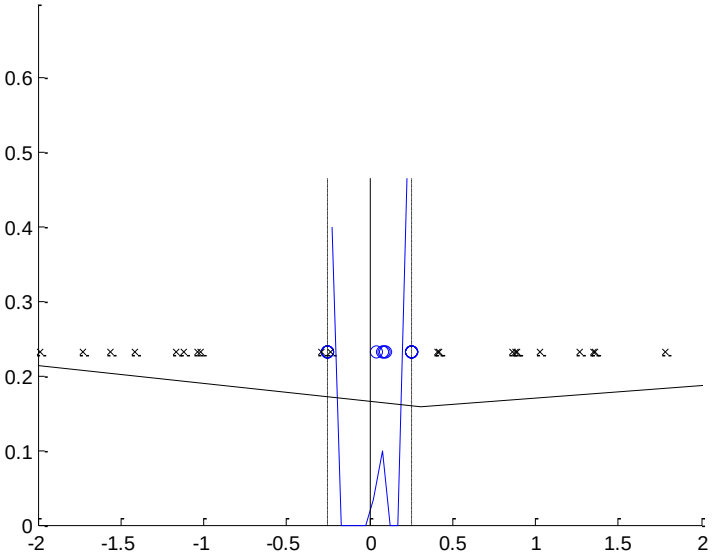
Figure 5.6 Histogram of residuals for training samples (in blue) and Universum samples of type 1 (in black) with no. of training samples = 30 and $\sigma = 0.5$

(a) histogram for standard SVM Regression model,(C= 12.06, $\varepsilon = 0.25$).

(b) histogram for U-SVM Regression model (C*/C=0.01, Δ=8).

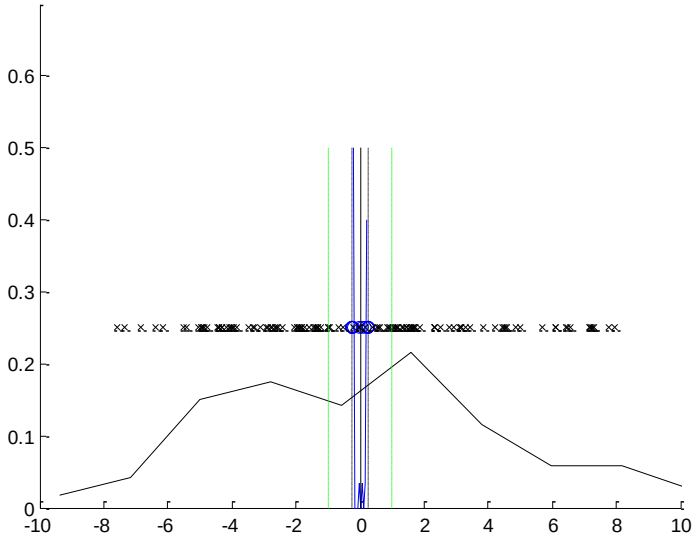


(a)

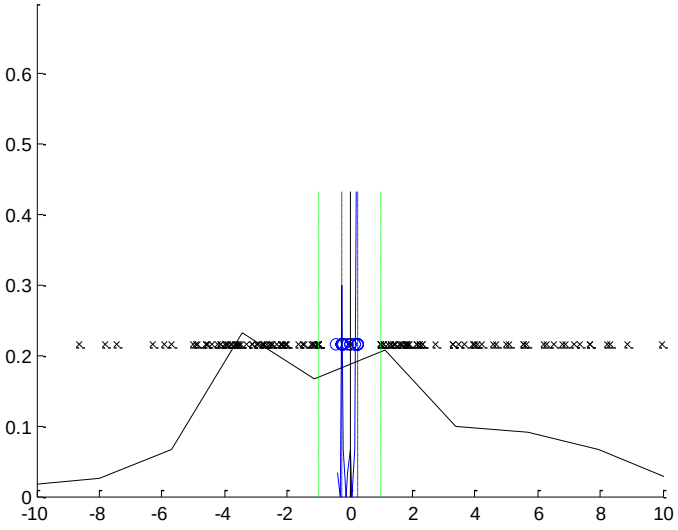


(b)

Figure 5.7 Histogram of residuals for training samples and Universum samples of type 1 with no. of training samples = 30 and $\sigma = 0.5$ (zoomed to a larger scale)
(a) histogram for standard SVM Regression model,(C= 12.06, $\varepsilon = 0.25$).
(b) histogram for U-SVM Regression model (C*/C=0.01, Δ=8).



(a)

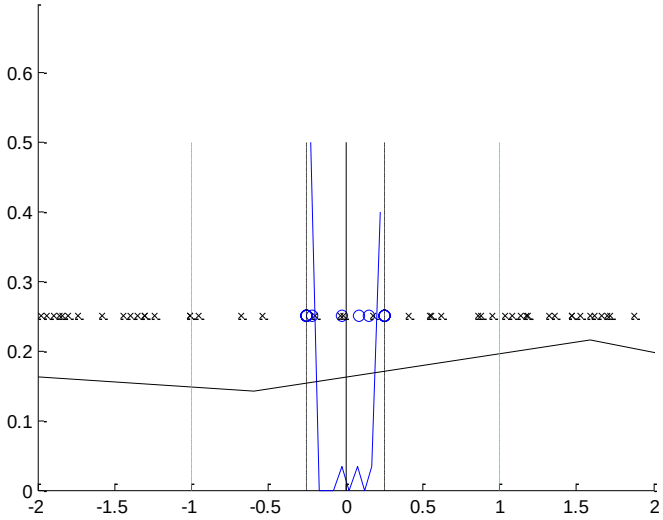


(b)

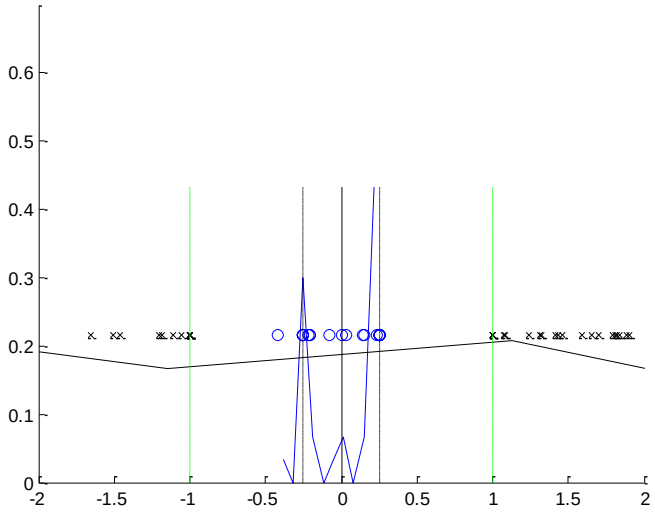
Figure 5.8 Histogram of residuals for training samples (in blue) and Universum samples of type 2 (in black) with no. of training samples = 30 and $\sigma = 0.5$

(a) histogram for standard SVM Regression model,(C= 12.06, $\varepsilon = 0.25$).

(b) histogram for U-SVM Regression model (C*/C=0.1, Δ=1).

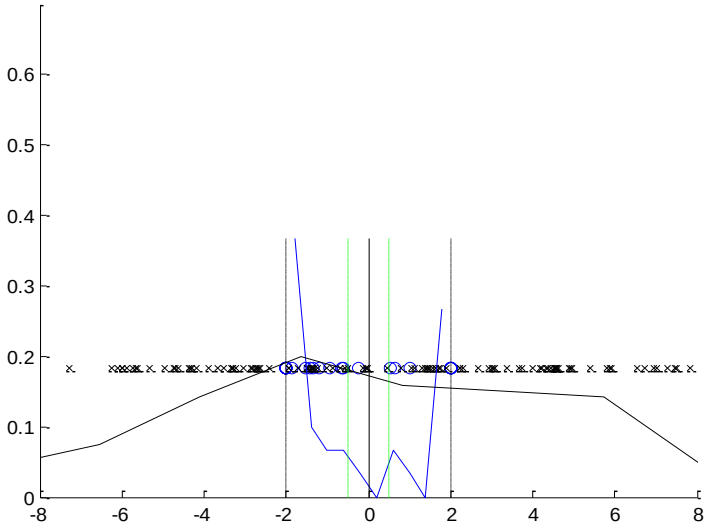


(a)

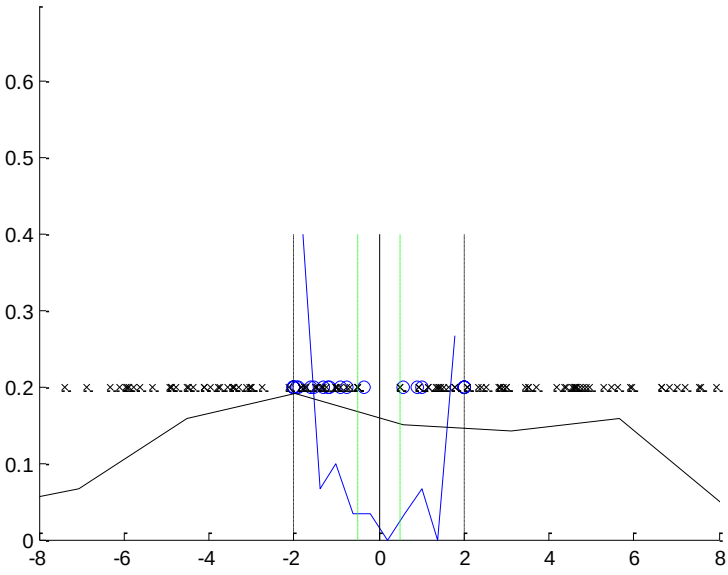


(b)

Figure 5.9 Histogram of residuals for training samples and Universum samples of type 2 with no. of training samples = 30 and $\sigma = 0.5$ (zoomed to a larger scale).
(a) histogram for standard SVM Regression model,(C= 12.06, $\varepsilon = 0.25$).
(b) histogram for U-SVM Regression model (C*/C=0.1, Δ=1).

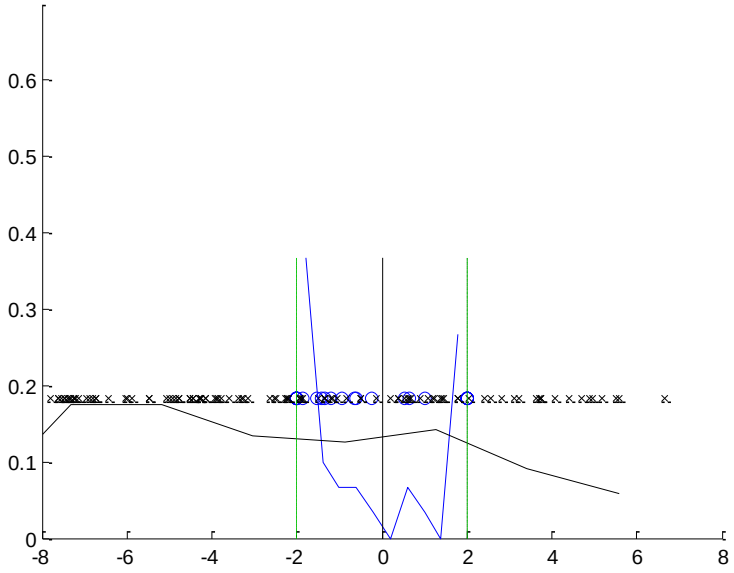


(a)

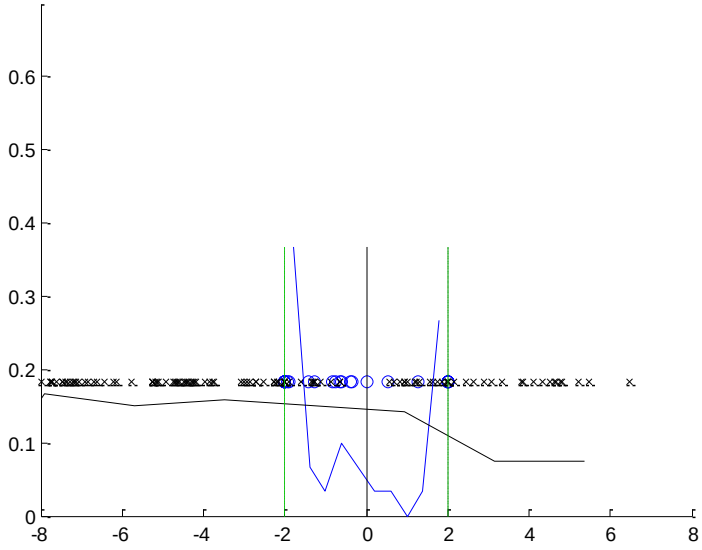


(b)

Figure 5.10 Histogram of residuals for training samples (in blue) and Universum samples of type 1 (in black) with no. of training samples = 30 and $\sigma = 2$
(a) histogram for standard SVM Regression model,(C= 13.59, $\varepsilon = 2$).
(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=0.5).



(a)

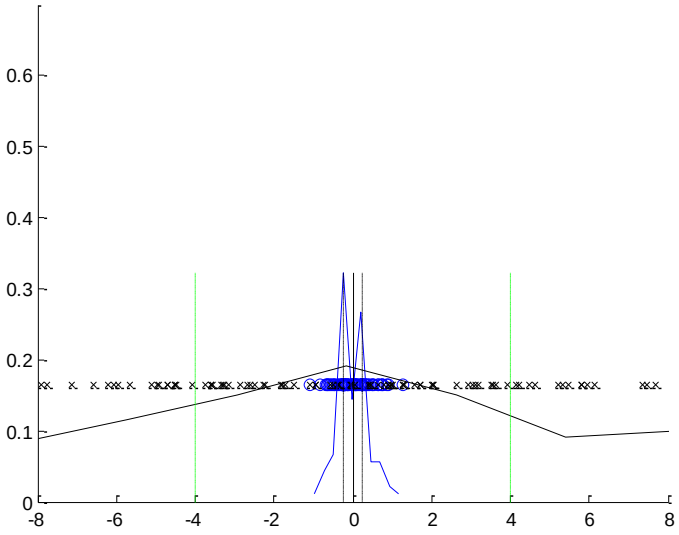


(b)

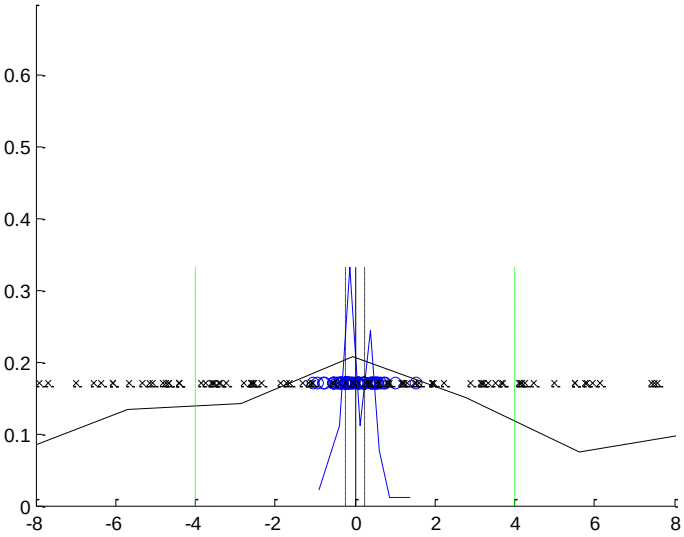
Figure 5.11 Histogram of residuals for training samples (in blue) and Universum samples of type 2 (in black) with no. of training samples = 30 and $\sigma = 2$

(a) histogram for standard SVM Regression model,(C= 13.59, $\varepsilon = 2$).

(b) histogram for U-SVM Regression model (C*/C=10^{-2}, Δ=2).

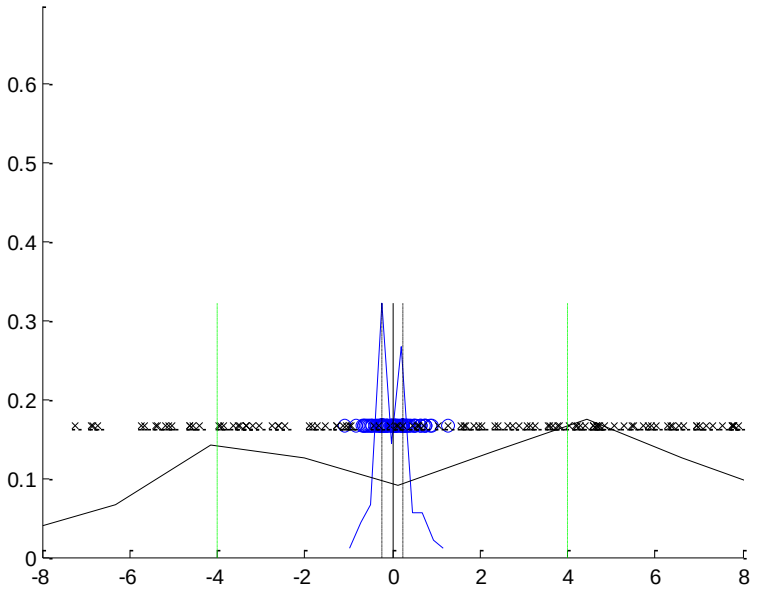


(a)

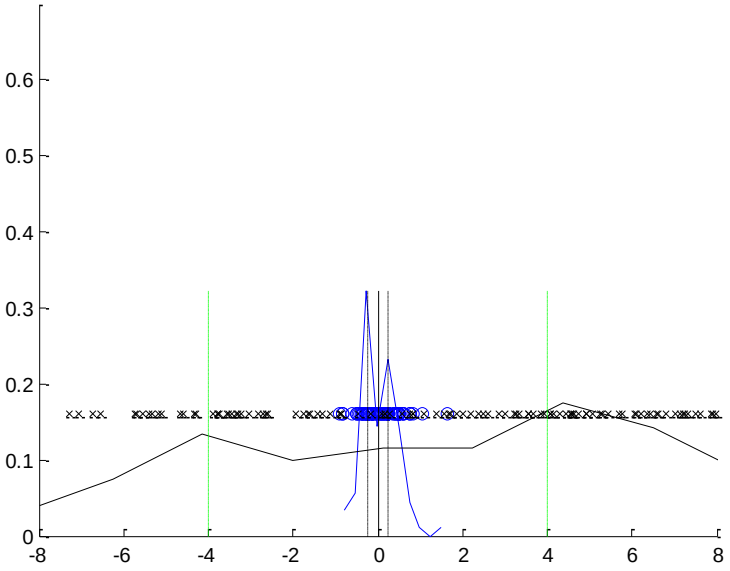


(b)

Figure 5.12 Histogram of residuals for training samples (in blue) and Universum samples of type 1 (in black) with no. of training samples = 90 and $\sigma = 0.5$
(a) histogram for standard SVM Regression model,(C= 14.38, $\varepsilon = 0.25$).
(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=4).



(a)

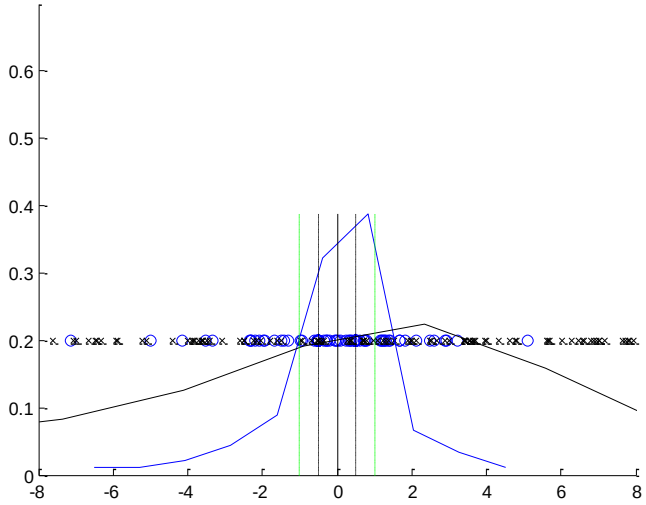


(b)

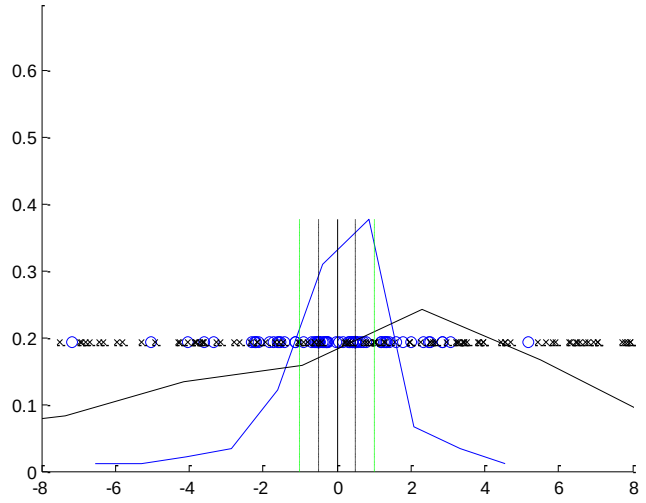
Figure 5.13 Histogram of residuals for training samples (in blue) and Universum samples of type 2 (in black) with no. of training samples = 90 and $\sigma = 0.5$

(a) histogram for standard SVM Regression model, (C= 14.38, $\varepsilon = 0.25$).

(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=4).



(a)

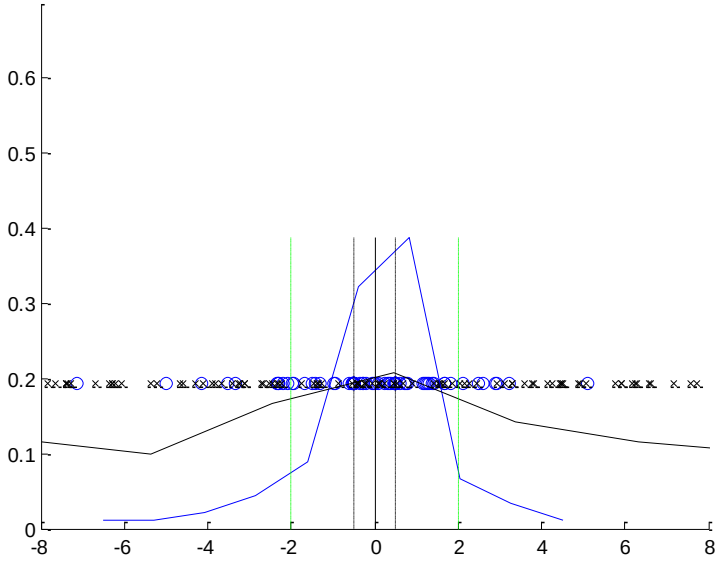


(b)

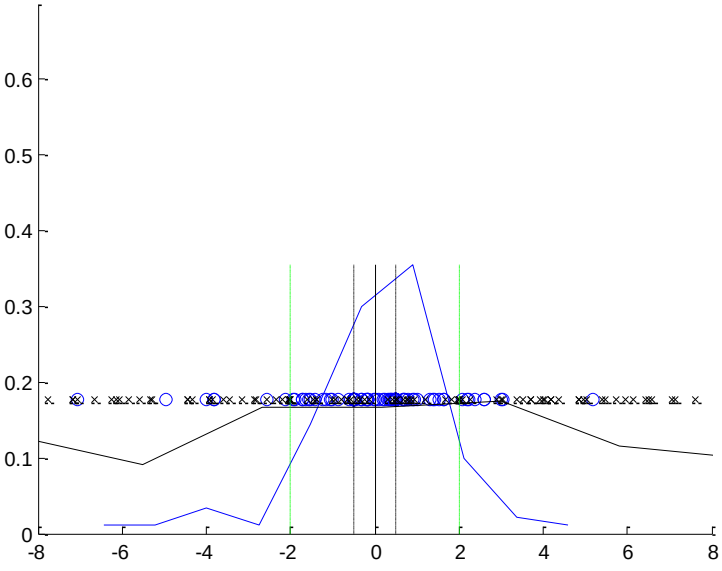
Figure 5.14 Histogram of residuals for training samples (in blue) and Universum samples of type 1 (in black) with no. of training samples = 90 and $\sigma = 2$

(a) histogram for standard SVM Regression model,(C= 15.4, $\varepsilon = 0.5$).

(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=1).

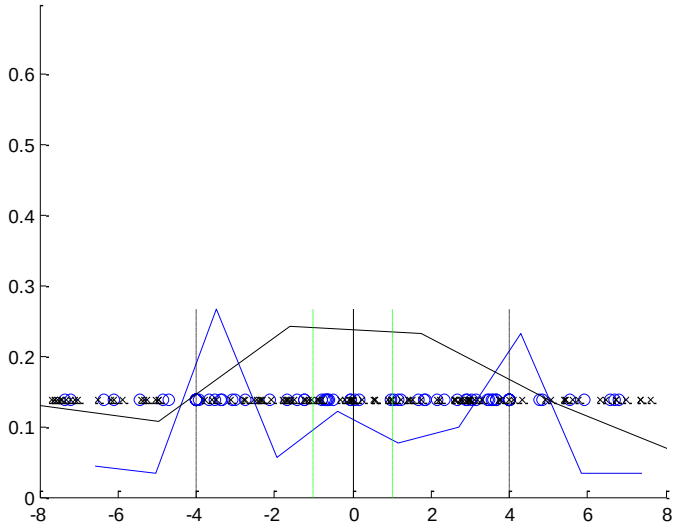


(a)

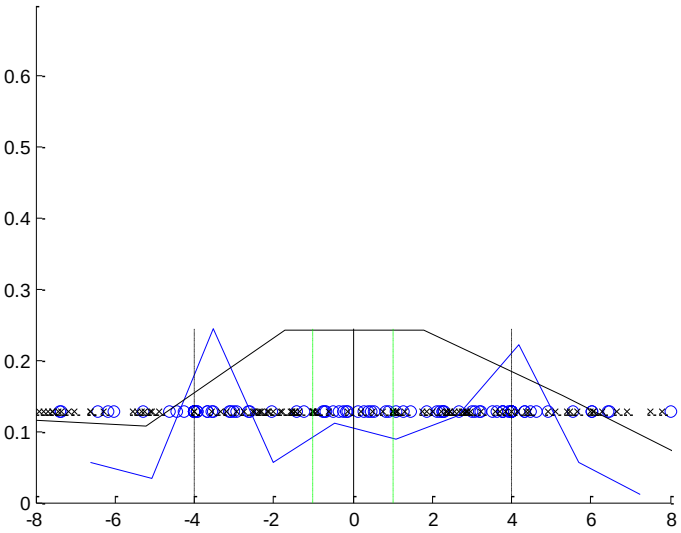


(b)

Figure 5.15 Histogram of residuals for training samples (in blue) and Universum samples of type 2 (in black) with no. of training samples = 90 and $\sigma = 2$
(a) histogram for standard SVM Regression model,(C= 15.4, $\varepsilon = 0.5$).
(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=2).

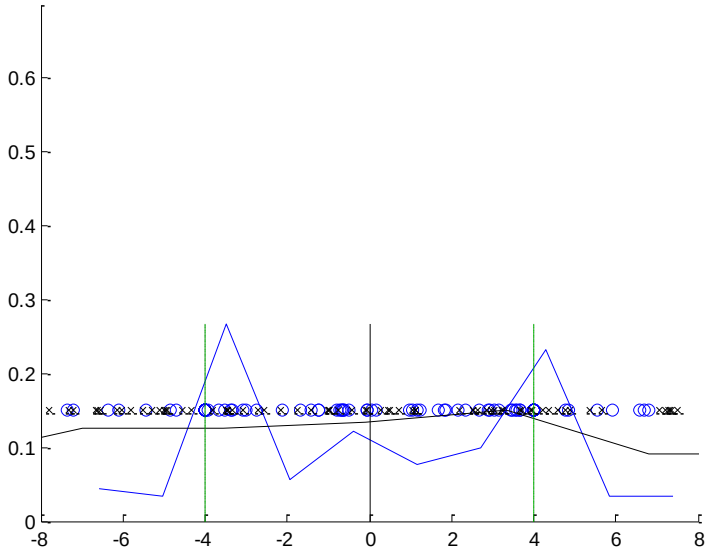


(a)

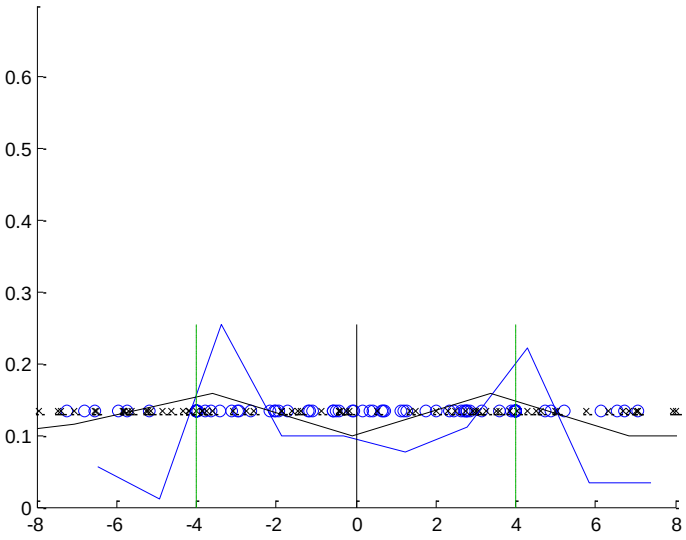


(b)

Figure 5.16 Histogram of residuals for training samples (in blue) and Universum samples of type 1 (in black) with no. of training samples = 90 and $\sigma = 4$
(a) histogram for standard SVM Regression model,(C= 25.29, $\varepsilon = 4$).
(b) histogram for U-SVM Regression model (C*/C=1, Δ=1).



(a)



(b)

Figure 5.17 Histogram of residuals for training samples (in blue) and Universum samples of type 2 (in black) with no. of training samples = 90 and $\sigma = 4$

(a) histogram for standard SVM Regression model,(C= 15.4, $\varepsilon = 4$).

(b) histogram for U-SVM Regression model (C*/C=10^{-1}, Δ=4).

Chapter 6 Summary and My Contributions

There are four major contributions described in this dissertation:

First, we developed the practical conditions for the effectiveness of Universum Learning for binary classification. We provide empirical results in support of our practical conditions.

Second, we provided an extension of Universum Learning to real life classification settings with different misclassification costs and unbalanced data. We also provided the practical conditions for the effectiveness of our proposed cost-sensitive U-SVM formulation over the cost-sensitive SVM and presented empirical results in support of our conditions.

Third, we extended Universum learning for single-class problems and provided the practical conditions for the effectiveness of our proposed single-class U-SVM formulation over the single-class SVM. We presented empirical results in support of our conditions.

Fourth, we provided an extension of Universum Learning to regression problems. We proposed the new U-SVM regression formulation and provided a CCCP based algorithm to solve it. Finally, we provided empirical comparisons between the proposed U-SVM and standard SVM regression.

There are still many open and challenging research questions. Many existing learning approaches assume a standard inductive learning formulation, where the goal is to estimate a predictive model from finite training data. While this inductive setting is

very general and commonly accepted, it cannot be taken for granted. As pointed out in (Cherkassky and Mulier, 2007), *future progress in predictive learning is likely to occur due to better understanding (and acceptance) of nonstandard learning inference, rather than marginal improvements of learning algorithms implementing standard inductive inference*. This line of thinking is in complete agreement with the work shown in this thesis, where we have demonstrated the advantages of a new learning methodology call Universum Learning compared to the methods based on standard inductive learning.

Publications from this thesis

V. Cherkassky, S. Dhar, and W. Dai, Practical Conditions for Effectiveness of the Universum Learning, *IEEE Transactions on Neural Networks*, vol.22, no. 8, pp. 1241-1255, Aug 2011.

S. Dhar, V. Cherkassky, Development and Evaluation of Cost-Sensitive Universum SVM, *IEEE Transactions on Systems, MAN, and Cybernetics PART B: Cybernetics*, Nov 2012, (minor revision).

V. Cherkassky, S. Dhar, Simple Method for Interpretation of High-Dimensional Nonlinear SVM Classification Models , *Proceedings of the 2010 International Conference on Data Mining*, July 2010.

S. Dhar, V. Cherkassky, Practical Analysis of the U-SVM Learning, *Snowbird Learning Workshop*, April 2011.

S. Dhar, and V. Cherkassky, Cost-Sensitive Universum-SVM, *ICMLA*, 2012.

S. Dhar, J. Lee, and V. Cherkassky, Single Class Universum-SVM, (submitted), preprint available on request. 2013.

S. Dhar, and V. Cherkassky, Universum-SVM for Regression, working paper, 2014.

References

Ahn, J., and Marron, J. S., The direction of maximal data piling in high dimensional space, *Technical Report*, University of North Carolina at Chapel Hill, 2005.

Bai, X. and Cherkassky, V. Gender classification of human faces using Inference through Contradictions, *Proc. IJCNN*,2008.

Bosch,A., Zisserman, A., Munoz,X., Representing shape with a spatial pyramid kernel, *CIVR*, Amsterdam , July 9–11, 2007.

Boyd, S. and Vandenberghe, L. , *Convex Optimization*, Cambridge University Press, 2004.

Cai, F.,Cherkassky, V.,Weisdorf, D., Arora, M., and Van Ness, B., Predictive modeling of Transplant-Related Mortality, *Proc. of the 2010 Design of Medical Devices Conf.*, Minneapolis, April 2010.

Camps-Valls, G., Rojo -Alvarez, J. L., and Martinez-Ramon, M. , *Kernel Methods in Bioengineering, Signal and Image Processing*. London: Idea Group Publishing, 2007.

Caruana, R.,Multi-task learning, *Machine Learning.*, vol. 28, pp. 41-75, July 1997.

Chandola, V., Banerjee, A. , and Kumar, V., Anomaly detection: A survey, *ACM Comput. Surv.*, vol. 41 , no. 3, pp. 1-58, 2009.

Chapelle, O., Schölkopf, B., and Zien, A., *Semi-Supervised Learning*, Cambridge. MA:The MIT Press, 2006.

Chen, S., and Zhang, C., Selecting Informative Universum Sample for Semi-Supervised Learning. *IJCAI*,2009.

Cherkassky, V., Alternative Formulations for Predictive Learning, Invited Talk, *International Conference on Artificial Neural Networks* (ICANN), Vienna, Austria, 2001

Cherkassky, V., and Dhar, S., Simple Method for Interpretation of High-Dimensional Nonlinear SVM Classification Models, *DMIN*, July 2010, pp. 267-272.

Cherkassky, V., *Predictive Learning*, www.VCtextbook.com, 2013.

Cherkassky, V. and Mulier, F., *Learning from Data Concepts: Theory and Methods*, 2nd ed. NY: Wiley, 2007.

Cherkassky, V. , Dhar, S. , and Dai, W., Practical Conditions for Effectiveness of the Universum Learning, *IEEE Transactions on Neural Networks*,vol.22, no. 8, pp. 1241-1255, Aug 2011.

Cherkassky, V. and Ma, Y. Practical Selection of SVM Parameters and Noise Estimation for SVM Regression, *Neural Networks*, 17(1), pp. 113-126, 2004.

Cherkassky, V. and Dai, W. , Empirical Study of the Universum SVM Learning for High-Dimensional Data," *in Proc. ICANN*, 2009.

Collobert, R., Sinz, F., Weston, J., and Bottou, L., "Large Scale Transductive SVMs," *JMLR*, vol 7 pp. 1687–1712, 2006.

Correa, R., Ludermir, T. , A quickly trainable hybrid SOM-based document organization system, *Neurocomputing* ,vol .71, pp. 3353-3359, 2008. doi:10.1016/j.neucom.2008.02.021.

Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A. and Vapnik, V. Support vector regression machines. *Advances in Neural Information Processing Systems*, 9:155–161, 1997.

Elkan, C., The foundations of cost-sensitive learning, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 2001.

Eskin, E., Arnold, A., Prerau, M., Portnoy, L., and Stolfo, S.,A geometric frame-work for unsupervised anomaly detection. *In Proceedings of Applications of Data Mining in Computer Security*, 2002.

Eskin, E., Lee, W., and Stolfo, S., Modeling system call for intrusion detection using dynamic window sizes, *In Proceedings of DISCEX*, 2001.

Evgeniou, T., and Pontil, M., Regularized multi-task learning, in *Proc. 17th SIGKDD Conf. on Knowledge Discovery and Data Mining*,2004, pp. 109-117.

Fanty, M., and Cole, R., Spoken letter recognition, *Advances in Neural Information Processing Systems 3*. San Mateo, CA: Morgan Kaufmann, 1991.

Gao, T. T., Yang, Z. X, Jing, L., On Universum-Support Vector Machines, *The Eighth International Symposium on Operations Research and Its Applications*, China, 2009, pp. 473-480.

Hastie, T., Tibshirani, R. and Friedman, J. , *The Elements of Statistical Learning. Data Mining, Inference and Prediction*. New York: Springer, 2001.

Heller, K. A., Svore, K.M., Keromytis, A., Stolfo, S.J., One class support vector machines for detecting anomalous windows registry accesses, *in: Proc. The workshop on Data Mining for Computer Security*, 2003, pp. 281-289.

Liang, L., and Cherkassky, V., Connection between SVM+ and Multi-Task Learning, *IJCNN*, 2008.

Lin, Y., Lee, Y. , and Wahba, G. , Support vector machines for classification in nonstandard situations, *Machine Learning*, vol. 46, pp. 191-202, 2002.

Manevitz, L. M., and Yousef, M., One-class SVMs for document classification, *Journal of Machine Learning Research*, vol. 2, pp. 139-154, 2002.

Roweis, S., sam roweis: data. [WWW page]. URL http://www.cs.nyu.edu/~roweis/data.html

Schölkopf, B., and Smola, A., *Learning with Kernels*. MIT Press, 2002.

Shen, C., Wang, P., Shen F., and Wang H., UBoost: Boosting with the Universum, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2011.

Shen, X., Tseng, G.C., Zhang, X., and Wong, W.H. , On (psi) Learning, *JASA*, vol 98 (463), pp. 724-734, 2003.

Sinz, F., Chapelle, O., Agarwal, A., and Schölkopf, B., An analysis of inference with the Universum, *In Proc. of 21st Annual Conference on Neural Information Processing Systems*, 2008, pp. 1-8.

Tan, P.N. , Steinbach, M. , and Kumar, V. , *Introduction to Data Mining*. New York: Pearson Education, 2006.

Vapnik, V. N., *Statistical Learning Theory*. New York: Wiley, 1998.

Vapnik, V. N.,*Estimation of Dependencies Based on Empirical Data. Empirical Inference Science:* Afterword of 2006. New York: Springer, 2006.

Weiss, G. M., McCarthy, K., and Zabar, B., Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?, *DMIN* 2007,pp. 35-41.

Weston, J., Collobert, R., Sinz,F., Bottou, L., and Vapnik, V., Inference with the Universum, *Proc. ICML*, 2006, pp. 1009-1016.

Ye, J., and Wang, T., Regularized (Quadratic) Discriminant Analysis for high dimensional, low sample size data, *Proc. SIGKDD*, 2006, pp 454—463.

Ye, J.,and Xiong, T., Computational and theoretical analysis of null space and orthogonal linear discriminant analysis, *Journal of Machine Learning Research*, vol. 7, pp. 1183—1204, 2006.

Zhang, D., Wang, J., Wang, F., and Zhang. C., Semi-Supervised Classification with Universum. *Proceedings of the 8th SIAM Conference on Data Mining (SDM)*, 2008, pp. 323-333.