2024-09-08

Distributionally robust binary classifier under Wasserstein distance

Huang, Qian

UNIVERSITY OF CALGARY

Distributionally Robust Binary Classifier Under Wasserstein Distance

by

Qian Huang

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN MATHEMATICS AND STATISTICS

CALGARY, ALBERTA

SEPTEMBER, 2024

Abstract

The robustification of statistical models has been a popular topic for decades. Statistical robustification and robust optimization are the two main approaches in the literature, where the former stabilizes the model output by removing the outlier points while the latter concerns more the outlier points in making the conservative decisions. This thesis develops a novel robust optimization perspective to robustify a class of binary classifiers. Our model considers the worst-case distribution within a pre-determined uncertainty ball that centers at the given benchmark distribution with the radius calculated as per the Wasserstein distance. We derive the tractable formulation for the general problem. When focusing on the support vector machine (SVM), the general problem boils down to an easy-to-solve second-order cone programming problem. The robustified SVM is then applied to synthetic data with and without contamination, and our simulation studies show that our robustified SVM model can outperform the classical SVM and the extreme empirical loss SVM models under many circumstances.

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my supervisors, Dr. Jingjing Wu and Dr. Qingrun Zhang, for their insightful guidance and unfailing patience throughout my master's program. Their immense knowledge and rigorous research mindset not only ensure the accomplishment of this project but also improve my problem-solving and other transferable skills.

I also would like thank my thesis defense examiners, Dr. Wenyuan Liao and Dr. Anatoliy Swishchuk, for reading the thesis and giving valuable suggestions. My thanks also go to the wonderful professors I had at University of Calgary who helped me to further my knowledge and passion in statistics. Moreover, I am thankful to the amazing support provided by the staff at the Department of Mathematics and Statistics (MTST). The financial support provided by MTST and MITACS Accelerate program are also very much appreciated.

Many thanks to Mike Newton and Chantelle Carly at 360 Engineering & Environmental Consulting Ltd. and Dr. Qingrun Zhang for the internship opportunity. The internship provides numerous avenues for implementing the statistical and data analytical skills that I learned from my master's program. Meeting and networking with the managers and colleagues also enhances my other soft skills.

Last but not least, I am truly indebted to my husband for his endless support throughout this journey. It is not possible to complete this program without his companion. I am also grateful to my parents for their understanding. The love from my family allows me to move forward without any concerns.

Table of Contents

List of Figures

List of Tables

List of Symbols, Abbreviations, and Nomenclature

Symbol	Definition
$\boldsymbol{X}$	The feature vector
$\boldsymbol{Y}$	The response variable
$\boldsymbol{x}_i$	The ith observed feature vector
y_i	The ith observed response
$\boldsymbol{\beta}$	The coefficient vector for the linear combination of features.
β_0	The intercept term
π_i	The prior probability for the ith class
L	The loss function
μ	The mean vector
Σ	The variance-covariance matrix
$\hat{F}_N$	The empirical cumulative distribution function
F	The cumulative distribution function
$W(\cdot, \cdot)$	The Wasserstein distance metric
R	The penalty function
C	The penalty coefficient/index
λ	The Lagrangian multiplier

EEL-SVM	The extreme empirical loss support vector machine
WD-SVM	The Wasserstein-distance-based support vector machine

Chapter 1

Introduction

The rapid growth of statistical learning has been witnessed in the past few decades. Two fundamental problems for statistical learning are regression and classification problems, where the former mainly concerns the numerical or quantitative response variables while the latter relates more with the categorical or qualitative response variables. Both problems have been extensively studied in the literature, with flourishing results achieved under the assistance of the fast-developing computing technologies. This thesis focuses on the classification problem. We introduce the background and review the classical classification models in the following two sections. This chapter is set out as follows. In Section 1.1, we look at the background for classification problems and the gap in the literature regarding the robustification of statistical models in the presence of data uncertainty. In Section 1.2, we review some classical models for the classification problems.

1.1 Background

The statistical learning models can be categorized into two basic streams – supervised learning and unsupervised learning models, where the former is often of more interest to the statisticians as these models investigate the connections between the features (also known as

the explanatory variables) and responses. Such connections can be of great scientific value as they probably reveal the law of nature.

Classification problems are quite common in practice. For example, in medical and biological sciences, a medical condition is often diagnosed based on the patient's symptoms; whether or not a person is subject to a certain cancer risk is often decided by her/his DNA sequence. In the industry of banking and investment, the predictions of stock price movement and fraudulent transactions are also classification problems. Within the realm of supervised learning, classical classification models include, to name but a few, (multinomial) logistic regression, Gaussian classifier, decision tree, support vector classifier (SVM) and artificial neural network (ANN). From the mathematical point of view, logistic regression and Gaussian classifier rely on more assumptions that support their applications. From the perspectives of flexibility and interpret-ability, the models which allow some kind of linear relationship between the features and responses, such as logistic regression, Gaussian classifier and SVM, enjoy more direct interpretations at the expense of flexibility, whereas the models based on the non-linear connections between the features and responses, such as decision tree and ANN, can be more flexibly applied but lose to some degree the interpret-ability.

Once the classification model is identified, the model parameters are usually estimated by using the likelihood-based, Bayesian or geometric methods. For example, the maximum likelihood estimation is used to estimate the parameters for the logistic regression; the Bayesian method is applied to find the decision boundary for the Gaussian classifier; the idea of optimal separating hyperplane is applied to locate the support vectors for SVM; the back-propagation is one main approach to estimate the parameters for ANN. As will be shown in the later chapters, for a variety of model fitting problems, the optimal set of parameters is the solution to an optimization problem, where the objective function comprises a sum of loss functions and a penalty term, which is used to alleviate the potential overfitting issues. In real-world applications, cross-validation is a frequently-used technique to examine the

validity/efficiency of a model or to determine the values of the tuning parameters. Cross-validation is developed from a re-sampling perspective, where in each round the data are partitioned into the training and testing sets, where the former is used for model fitting and the latter is used for out-of-sample testing. A n-fold cross-validation involves n rounds of model fitting.

In almost all the classical statistical models, the model is fitted by using the observed data only, though re-sampled data are also used in a few cases. Due to the pivotal role played by the model fitting, it is crucial to examine the credibility (or simply understood as the quantity and quality) of the data being used for model fitting. For many real life problems, the data credibility cannot be warranted, which is partially due to, for example, sampling error, data noise, and accessibility of some public sources. This inevitably gives rise to the so-called data uncertainty issue. The presence of data uncertainty can bring very adverse consequences to the descriptive and predictive capacities of the fitted model, since a small change of the input data can lead to significantly different model output. As such, it is very common to robustify the models before putting them in use, so that the model output is not quite sensitive to the small perturbation of the input data.

The robustification of a statistical model can be achieved through different manners, where the two main frameworks are robust statistics and robust optimization. These two frameworks yield different notions of robustness. The first kind of robustness, which stems from the vast literature on robust statistics, mainly studies the change of an estimator with respect to the change of data. To fairly compare the changes of different estimators, the influence function approach, which is proposed by Hampel (1974), is quite often used. Under the first notion of robustness, the common way to robustify a statistical model is to eliminate the influence of outlier points. For example, Value-at-Risk (VaR) is more robust than Expected Shorfall (ES), since the former is simply a quantile while the latter is the tail average. Such robustification requires modifying the loss function being used for deriving the parameters. For example, for SVM Wu and Liu (2007) proposed using the truncated

hinge loss function instead of the classical hinge loss function to limit the bias effect of the outlier points on the optimization results. In sharp contrast to robust statistics, robust optimization applies the well-known 'min-max' approach in operational research, leading to the parameter estimation based on the worst-case distribution. In other words, robust optimization would amplify the influence of outlier points and build the model from the most conservative perspective. The interested readers are referred to, to name but a few, Xu et al. (2009), Lee and Mehrotra (2015), Asimit et al. (2022) and the references cited therein.

In this thesis, we are exploring the robustification of a class of classification models within the framework of robust optimization. We first review some of the classical classification models in the next section.

1.2 The classical classification models

As a generic setting, we denote the response variable by Y and the feature vector by $\boldsymbol{X} = (X_1, \ldots, X_n)$. The sample points are denoted by $\{(y_i, \boldsymbol{x}_i)\}$, where y_i denotes the observed response and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{in})$ denotes the observed feature vector. We use $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)$ to denote the linear coefficients of the features if the model involves the linear combination of the features.

1.2.1 Logistic regression

In a logistic regression, the response variable Y is a categorical variable and can only take $K \in \mathbb{Z}^+$ classes. The probabilities of Y taking these K classes are modelled via the logistic function, which connects the log odds with the linear combination of the features, i.e.

$$\ln \frac{\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x})}{\mathbb{P}(Y = K|\boldsymbol{X} = \boldsymbol{x})} = \boldsymbol{\beta}_i^T \boldsymbol{x} \tag{1.1}$$

for $i = 1, 2, \ldots, K$. Though the equation (1.1) uses the Kth class as the reference class, the fitted model remains the same if any other class is used as the reference class. After simple

algebraic manipulations, one can derive the following from the equation (1.1)

$$\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x}) = \frac{\exp(\boldsymbol{\beta}_i^T \boldsymbol{x})}{1 + \sum_{k=1}^{K-1} \exp(\boldsymbol{\beta}_k^T \boldsymbol{x})} \tag{1.2}$$

for $i = 1, 2, \ldots, K-1$, and

$$\mathbb{P}(Y = K|\boldsymbol{X} = \boldsymbol{x}) = \frac{1}{1 + \sum_{k=1}^{K-1} \exp(\boldsymbol{\beta}_k^T \boldsymbol{x})}.$$

In practice, applying the logistic regression to binary responses occurs quite often. For example, the result is either positive or negative for a disease screening test, a patient either survives or dies after receiving some particular kind of medical treatment, the contamination is either present or absent for a oil & gas field investigation. For a binary response case, assuming that Y can only take 0 and 1, the log-likelihood function for the logistic regression model is

$$\begin{aligned} l(\boldsymbol{\beta}) &= \sum_{i=1}^{N} \{y_i \ln \mathbb{P}(Y = 1|\boldsymbol{X} = \boldsymbol{x}_i) + (1 - y_i) \ln \mathbb{P}(Y = 0|\boldsymbol{X} = \boldsymbol{x}_i)\} \\ &= \sum_{i=1}^{N} \left\{y_i \boldsymbol{\beta}^T \boldsymbol{x}_i - \ln(1 + \exp(\boldsymbol{\beta}^T \boldsymbol{x}_i))\right\}. \end{aligned}$$

Maximizing the log-likelihood function is an optimization problem, where plenty of efficient numerical methods are readily available, such as (stochastic) gradient descent method that can be applied to the maximization of $l(\boldsymbol{\beta})$ directly, or the Newton-Raphson method that is used to seek the roots of $l'(\boldsymbol{\beta}) = 0$. The interested readers are referred to, for example, Goodfellow et al. (2016) for a detailed review of these methods.

1.2.2 Gaussian classifier

The Gaussian classifier is also known as the linear discriminant analysis (LDA) method in statistical learning, which calculates the posterior probabilities of Y taking the K classes

from Bayesian lens:

$$\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x}) = \frac{f_i(\boldsymbol{x})\mathbb{P}(Y = i)}{\sum_{k=1}^{K} f_k(\boldsymbol{x})\mathbb{P}(Y = k)}, \tag{1.3}$$

where $f_k(\cdot)$ denotes the density function of $\boldsymbol{X}$.

The key assumption underlying the Gaussian classifier is that the feature vector $\boldsymbol{X}$ conditionally follows a multi-normal distribution, i.e.

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma_k|^{1/2}} \exp(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T\Sigma_k^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)), \tag{1.4}$$

where μ_k is the mean vector and Σ_k is the variance-covariance matrix of $\boldsymbol{X}|Y = k$. Letting $\pi_i = \mathbb{P}(Y = i)$ be the prior probability, if Σ is the common variance-covaraince matrix across the various classes, i.e. $\Sigma_k = \Sigma$, the log ratio of the posterior probabilities $\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x})$ and $\mathbb{P}(Y = j|\boldsymbol{X} = \boldsymbol{x})$ is

$$\ln \frac{\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x})}{\mathbb{P}(Y = j|\boldsymbol{X} = \boldsymbol{x})} = \ln \frac{\pi_i}{\pi_j} - \frac{1}{2}(\mu_i + \mu_j)^T\Sigma^{-1}(\mu_i - \mu_j) + \boldsymbol{x}^T\Sigma^{-1}(\mu_i - \mu_j),$$

which is a linear function of x. Note that $\mathbb{P}(Y = i|\boldsymbol{X} = \boldsymbol{x}) = \mathbb{P}(Y = j|\boldsymbol{X} = \boldsymbol{x})$ leads to

$$\ln \frac{\pi_i}{\pi_j} - \frac{1}{2}(\mu_i + \mu_j)^T\Sigma^{-1}(\mu_i - \mu_j) + \boldsymbol{x}^T\Sigma^{-1}(\mu_i - \mu_j) = 0, \tag{1.5}$$

which is a hyperplane that separates the feature space. The equation (1.5) is called the Bayesian decision boundary between the classes i and j.

The assumption of common variance-covariance matrix can easily fail to hold in practice. When different variance-covariance matrices are applied, the Bayesian decision boundary as computed by $\ln \frac{\mathbb{P}(Y=i|\boldsymbol{X}=\boldsymbol{x})}{\mathbb{P}(Y=j|\boldsymbol{X}=\boldsymbol{x})} = 0$ can be shown to be of the quadratic equation of $\boldsymbol{x}$. The Gaussian classifier in the absence of the common variance-covariance matrix assumption is also named as the quadratic discriminant analysis method. More details regarding the Gaussian classifier can be found in Hastie et al. (2009).

For real-life problems, the assumption of multi-normally distributed $\boldsymbol{X}$ can be tested by,

for example, Mardia's test, and the assumption of common variance-covariance matrix can be tested by using Bartlett's test or Levene's test. Further details are not pursued in this thesis.

1.2.3 Support vector classifier

The idea of support vector classifier stems from the construction of the optimal separating hyperplane for the perfectly separable classes, with extension to the non-separable classes. Though in practice multiple hyperplanes can be constructed to separate the feature space into several non-overlapping regions for the classification purpose when there are multiple classes, we focus on the binary case where only one hyperplane is to be sought. Different from Section 1.2.1, the two classes in this section are encoded as 1 and -1.

Recall the elementary geometry that the signed distance from any point $\boldsymbol{x}$ to a hyperplane $\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0 = 0$ is given by $\frac{1}{||\boldsymbol{\beta}||}(\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0)$. A natural choice of the classifier is thus $\text{sign}(\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0)$, where $y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0) > 0$ if the ith point is located within the correct class. The uniqueness of such hyperplane cannot be guaranteed for the perfectly separable classes, which motivates the search of the hyperplane that creates the largest margin between the training points for different classes:

$$\begin{cases} \max\limits_{\boldsymbol{\beta},\beta_0,M} M, \\ \quad \text{s.t. } \dfrac{y_i}{||\boldsymbol{\beta}||}(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0) \geq M, \text{ for } i = 1, 2, \ldots, N, \end{cases} \tag{1.6}$$

where M denotes the margin between the nearest point and the separating hyperplane. Note that the hyperplane remains the same if $\boldsymbol{\beta}$ and β_0 are re-scaled by the same magnitude. As such, it can be easily shown that the model (1.6) is equivalent to

$$\begin{cases} \min\limits_{\boldsymbol{\beta},\beta_0} ||\boldsymbol{\beta}||, \\ \text{s.t. } y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0) \geq 1 \text{ for } i = 1, 2, \ldots, N. \end{cases} \tag{1.7}$$

The hyperplane resulting from the model (1.7) can be applied to the perfectly separable classes. However, the solution to (1.7) does not exist for the non-separable classes. To find the hyperplane that allows for some points to be on the wrong side of the margin, the model (1.6) or (1.7) needs to incorporate more flexibility. A straightforward way is to "soften" the margin such that the constraint of (1.6) becomes

$$y_i(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0) \geq M(1 - \xi_i) \text{ for } i = 1, 2, \ldots, N$$

with $\xi_i \geq 0$ and $\sum_{i=1}^{N} \xi_i \leq C$ for some constant $C \geq 0$. Intuitively, by introducing the extra parameter ξ_i, the ith sample point is allowed to be on the wrong side of the margin and thus misclassified. The constraint $\sum_{i=1}^{N} \xi_i \leq C$ places a constraint on the number of such misclassifications. With the new constraint, the model (1.7) becomes

$$\begin{cases} \min\limits_{\boldsymbol{\beta}, \beta_0, \boldsymbol{\xi}} ||\boldsymbol{\beta}||, \\ \quad \text{s.t. } y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) \geq 1 - \xi_i \text{ for } i = 1, 2, \ldots, N, \\ \qquad \xi_i \geq 0, \ \sum\limits_{i=1}^{N} \xi_i \leq C, \end{cases} \tag{1.8}$$

where $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)$. Note that the objective function is convex in $\boldsymbol{\beta}$ and all the constraints are linear, the duality gap is closed due to the Slater's condition (Boyd and Vandenberghe, 2004). Therefore, solving the problem (1.8) is equivalent to solving

$$\begin{cases} \min\limits_{\boldsymbol{\beta}, \beta_0, \boldsymbol{\xi}} ||\boldsymbol{\beta}|| + \lambda \sum\limits_{i=1}^{N} \xi_i, \\ \quad \text{s.t. } \xi_i \geq 1 - y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) \text{ and } \xi_i \geq 0 \text{ for } i = 1, 2, \ldots, N, \end{cases} \tag{1.9}$$

with $\lambda \geq 0$ being the Lagrangian parameter. By simple variable substitutions, the above

problem can be further shown to be equivalent to

$$\min_{\boldsymbol{\beta},\beta_0} \ ||\boldsymbol{\beta}|| + \lambda \sum_{i=1}^{N} (1 - y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0))_+, \tag{1.10}$$

where $(1-z)_+ = \max\{1-z, 0\}$ is called the hinge loss function. The formulation (1.10) can be understood as the sum of the aggregate loss $\sum_{i=1}^{N}(1 - y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0))_+$ and the penalty term $||\boldsymbol{\beta}||$, where the latter is used to regularize the size of $\boldsymbol{\beta}$ to alleviate the overfitting issues. The Lagrangian parameter λ is a coefficient indicating the relative importance of the aggregate sum in the whole problem.

The closed-form solution to the problem (1.10) exists, and we refer the interested readers to Hastie et al. (2009) for the detailed derivations. In practice, one can also apply any convex programming package to solve the problem (1.8) as it is a classical linearly-constrained-quadratic-programming (LCQP) problem. We do not pursue the computational details further in this thesis.

1.3 Goals and structure of the thesis

As shown in Section 1.2, the classical statistical models are built on the observed data without considering any data uncertainty. This is tantamount to using the empirical distribution in model fitting. The empirical distribution can however be quite different when the data sets are of different sizes, which lowers the reliability of empirical distribution in practice. The following chapters of this thesis aim to address this issue by using the robust versions of the well-known statistical models. The rest of this thesis is structured as follows. Chapter 2 reviews the existent robust SVM models from both the statistical robustness and robust optimization perspectives. Chapter 3 reviews the popular Wasserstein distance (WD) and builds the robust SVM model by considering the worst-case distribution within a Wasserstein ball. A tractable formulation for the WD-SVM model is derived. Chapter 4 compares the WD-SVM model with the classical SVM and another robust optimization SVM models by

applying them to synthetic data sets without and with data uncertainty. Chapter 5 concludes the paper and presents several future research directions.

Chapter 2

Robust classifiers

As mentioned in Chapter 1, many statistical models are sensitive to the outlier points. Recall the soft-margin support vector binary classifier, which is a hyperplane that does not perfectly separate the two classes whose linear coefficients are derived from the optimization problem (1.10), the aggregate loss $\sum_{i=1}^{N}(1 - y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0))_+$ can be significantly influenced by a few points which are on the wrong side of the margin and whose $y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) << 0$[1]. To mitigate the influence of the outlier points, modification of the loss function is often used in the literature, and this vein of robustification is called statistical robustification. In contrast to that, to be more conservative in choosing the classifier, a decision maker may exaggerate the influence of the outlier points, which yields the model that over-weighs the outlier points. Such vein of robustification is within the realm of robust optimization. In this chapter we review some of the prevailing models that are situated in these two veins. As this thesis does not pursue a comprehensive review of the robustification of all the statistical models, we will focus on the robusitification of SVM only in the remaining of this chapter.

[1] Here we use the notation '<<' to denote "much smaller than".

2.1 Statistically robust classifiers

It is easily noted that the sensitivity of the classical SVM with respect to the outlier points can be ascribed to the unbounded hinge loss function, i.e. $\lim_{z\to-\infty}(1-z)_+ = \infty$. The common ways handling the sensitivity include the modifications of the loss function and the use of weighted sum in computing the aggregate loss.

2.1.1 Truncated hinge loss

Wu and Liu (2007) proposed using the following truncated hinge loss function to replace the classical hinge loss function:

$$l_T(z) = (1-z)_+ - (s-z)_+ = \begin{cases} 0, & z \geq 1, \\ 1-z, & 1 > z \geq s, \\ 1-s, & s > z. \end{cases} \tag{2.1}$$

The truncated hinge loss function is also named as the ramp loss function if $s = -1$ (Brooks, 2011). By using the truncated hinge loss function, the bias effects of the outlier points are limited. Furthermore, Wu and Liu (2007) showed that the truncated hinge loss function is fisher consistent (or classification calibrated), which is a desirable feature for a loss function. We will review more details of this property in Chapter 3. Despite the intuition behind the use of truncated hinge loss function, the resultant optimization problem

$$\min_{\boldsymbol{\beta},\beta_0} \ ||\boldsymbol{\beta}|| + \lambda \sum_{i=1}^{N} l_T(y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)) \tag{2.2}$$

is no longer a convex programming problem, which gives rise to non-trivial challenges for optimization.

In the literature, there are two main approaches to solve the optimization problem (2.2). Note that the truncated hinge loss function can be written as the difference between two

convex functions, thus the use of difference-of-convex algorithm (DCA) is a natural way to find the optimum of (2.2). The DCA solves the problem (2.2) by minimizing a sequence of convex subproblems, and the found optimum is dependent on the initial point and can therefore be a local optimum. As such, one needs to randomize the initial points and run the DCA multiple times to get the optimum point that is close enough to the global optimum. More details regarding the DCA can be found in, for example, Liu et al. (2005) and the references cited therein.

Instead of using the DCA, Brooks (2011) proposed the use of integer programming to handle the ramp loss function. With the ramp loss function, $l_T(z) = (1-z)_+ - (-1-z)_+ = (1-z)_+ \wedge 2$. Similar to Section 1.2.3, the problem (2.2) can be written as

$$\begin{cases} \min\limits_{\boldsymbol{\beta},\beta_0,\boldsymbol{\xi},\boldsymbol{z}} \ ||\boldsymbol{\beta}|| + \lambda(\sum\limits_{i=1}^{N}\xi_i + 2\sum\limits_{i=1}^{N} z_i), \\ \quad \text{s.t. } y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0) \geq 1 - \xi_i \text{ if } z_i = 0 \text{ for } i = 1, 2, \ldots, N, \\ \qquad z_i \in \{0, 1\} \text{ for } i = 1, 2, \ldots, N, \\ \qquad \xi_i \in [0, 2] \text{ for } i = 1, 2, \ldots, N. \end{cases} \tag{2.3}$$

The above integer programming problem is solved by using IBM ILOG CPLEX optimization studio, where a branching scheme for branching on disjunctions such as the indicator constraints is implemented, and among the generated solutions the best-known solution is used if the provable optimality is not obtained after certain CPU seconds. Such practice is also known as the stopping heuristic.

2.1.2 Hard margin loss

The hard margin loss function is another loss function being applied in Brooks (2011) and has the form

$$l_H(z) = \mathbb{1}_{(-\infty,1]}(z). \tag{2.4}$$

In other words, no loss is incurred if $y_i(\boldsymbol{\beta}_T \boldsymbol{x}_i + \beta_0) > 1$, while one unit of loss is incurred by the ith sample point if $y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) \leq 1$. Minimizing the aggregate loss amounts to minimizing the number of sample points that are located within the margin or misclassified. Figure 2.1 shows the comparison of three loss functions.

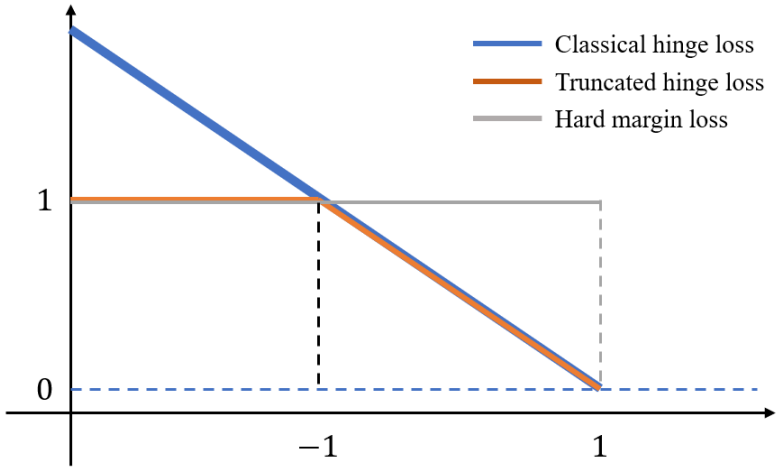


Figure 2.1: The comparison of the three loss functions.

With the hard margin loss function, the optimization problem is turned into

$$\begin{cases} \min\limits_{\boldsymbol{\beta},\beta_0,\boldsymbol{z}} \ ||\boldsymbol{\beta}|| + \lambda \sum\limits_{i=1}^{N} z_i, \\ \text{s.t. } y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) \geq 1 \text{ if } z_i = 0 \text{ for } i = 1, 2, \ldots, N, \\ \quad z_i \in \{0, 1\} \text{ for } i = 1, 2, \ldots, N. \end{cases} \tag{2.5}$$

Similar to the problem (2.3), the above optimization problem is an integer programming problem, which can be solved using a similar manner. Brooks (2011) conducted two sets of simulation analyses and showed that the SVM with either the ramp or hard margin loss function can approximate the Bayesian optimal classifier reasonably well in the presence of data contamination. However, for the eleven real-life datasets, the SVM with the ramp loss function generally outperforms that with the hard margin loss function.

2.1.3 Adaptively weighted classifier

Recall that in the classical SVM, the losses $\{(1-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))_+\}_{i=1}^N$ are assigned the equal weights in the summation. To mitigate the influence of the outlier points, one can reduce the weights for the large losses. Thus, Wu and Liu (2013) proposed re-weighting the losses to lower the influence of the outlier points.

Straightforwardly, choosing appropriate weights plays a crucial role in the implementation of such methodology. In the one-step weighted SVM proposed by Wu and Liu (2013), the weight for the ith sample point is chosen to be

$$w_i = \frac{1}{1+|\hat{f}_{SVM}(\boldsymbol{x}_i)|}, \tag{2.6}$$

where $\hat{f}_{SVM}(\boldsymbol{x}) = \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T\boldsymbol{x}$ is the classifier function resulting from the solution to the problem (1.10). Note that given a large $|\hat{f}_{SVM}(\boldsymbol{x}_i)|$ the ith sample point can either be misclassified or correctly classified. In the former case, the large $|\hat{f}_{SVM}(\boldsymbol{x}_i)|$ indicates that the ith sample point is nearly an outlier point, which needs to be assigned a small weight. If the ith sample point is correctly classified, the large $|\hat{f}_{SVM}(\boldsymbol{x}_i)|$ implies that the ith sample point is almost unlikely to be located on the margin or within the support vectors, which, as per the solution strucutre, further implies that the ith sample point does not contribute to the classifier function. From this perspective, assigning the ith sample point a small weight also makes sense. By re-weighting the losses using the weights (2.6), the problem (1.10) becomes

$$\min_{\boldsymbol{\beta},\beta_0} \; ||\boldsymbol{\beta}|| + \lambda\sum_{i=1}^{N} w_i(1-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))_+, \tag{2.7}$$

which is again a convex optimization problem, and can be solved by using the readily available convex programming packages.

Besides the one-step weighted SVM, Wu and Liu (2013) proposed another iterative weighted SVM, where the ith sample point is assigned the weight 1 if $y_i\hat{f}^{(t)}_{SVM}(\boldsymbol{x}_i) \in [s,1]$.

Here $\hat{f}_{SVM}^{(t)} = \hat{\beta}_0^{(t)} + \boldsymbol{\beta}^{(t)^T}\boldsymbol{x}$ is the classifier function resulting from the solution to the problem

$$\min_{\boldsymbol{\beta},\beta_0} \ ||\boldsymbol{\beta}|| + \lambda \sum_{i=1}^{N} w_i^{(t)}(1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0))_+, \tag{2.8}$$

where $\{w_i^{(t)}\}_{i=1}^N$ are the weights in the tth iteration. The iteration stops when $w_i^{(t)} = w_i^{(t-1)}$ for $i = 1, 2, \ldots, N$. Wu and Liu (2013) also proved that the iterative weighted SVM coincides with one local solution for the problem (2.2), which bridges these two different robustification methods and provides another perspective to understand the robustification mechanism of the iterative weighted SVM.

2.2 Robust-optimization-based robust classifiers

The core idea inside of RO is "robust to disturbance", which refers to the stable model output under the small change of model input. The data disturbance is common and can occur at any stages of data collection or generation. For example, vague tones are often detected in sentimental analyses of text; the precision of data measurement might be varying from time to time; the data might even be manipulated to decrease the misclassification rate. Considering data disturbances helps making more conservative decisions, which in practice can avoid significantly negative consequences. The RO has a long history in a variety of fields. For example, the well-known Markowitz portfolio optimization theory was developed by Blanchet et al. (2022) where the distributional uncertainties regarding the distributions of return rates of different assets are incorporated. Boonen and Jiang (2024) applied the RO in seeking the optimal insurance contract from the policyholder's perspective. In the remaining of this section, we review some applications of the RO in robustifying the classification models.

2.2.1 Box-type uncertainty

Among the early works on applying the RO in classification model robustification, Xu et al. (2009) proposed the use of the disturbed data instead of the observed data in the problem (1.10). Due to the uncertainties of the disturbances, the model is fitted based on the worst-case disturbances.

Let $\boldsymbol{\delta}_i \in \mathbb{R}^n$ be the disturbance of the ith sample point, the model considered by Xu et al. (2009) is of the form

$$\min_{\boldsymbol{\beta},\beta_0} \max_{\boldsymbol{\delta}_1,\ldots,\boldsymbol{\delta}_N \in \mathcal{D}} ||\boldsymbol{\beta}|| + \lambda \sum_{i=1}^{N} (1 - y_i(\boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{\delta}_i) + \beta_0))_+, \tag{2.9}$$

where $\mathcal{D}$ is a box-type uncertainty set. A often-used box-type uncertainty set is $\mathcal{D} = \left\{\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_N \in \mathbb{R}^n : \sum_{i=1}^N ||\boldsymbol{\delta}_i|| \leq c\right\}$ for some $c \in \mathbb{R}^+$. An important result achieved in Xu et al. (2009) is the equivalence between the problem (2.9) and the following problem

$$\min_{\boldsymbol{\beta},\beta_0} (1 + c)||\boldsymbol{\beta}|| + \lambda \sum_{i=1}^{N} (1 - y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0))_+, \tag{2.10}$$

which turns the RO problem (2.9) into a similar type regularized optimization problem of (1.10). Intuitively, $c = 0$ corresponds to no disturbances, and the problem (2.10) reduces to (1.10). For a larger c, the regularization of (2.10) on $\boldsymbol{\beta}$ is stronger, leading to a more robust model.

Besides establishing the equivalence between the problems (2.9) and (2.10), Xu et al. (2009) also showed the use of the solutions of (2.9) under different c to approximate the following chance-constrained classifier:

$$\begin{cases} \min\limits_{\boldsymbol{\beta},\beta_0,\mathcal{L}} \mathcal{L}, \\ \quad \text{s.t. } \mathbb{P}(\sum\limits_{i=1}^{N} (1 - y_i(\boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{\delta}_i) + \beta_0))_+ \leq \mathcal{L}) \geq 1 - \alpha, \end{cases} \tag{2.11}$$

where the disturbances $(\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_N)$ are assumed to follow a joint distribution and $\alpha \in (0, 1)$ denotes the confidence level. Note that (2.11) does not consider the worst-case disturbances. The approximation adds probabilistic interpretations for the SVM under box-type uncertainty.

2.2.2 Extreme empirical loss

Value-at-Risk (VaR) and Tail Value-at-Risk (TVaR) are two prominent risk measures in risk management field, whose definitions are respectively given by

$$\mathrm{VaR}_\alpha(X) = \inf\{x : \mathbb{P}(X \leq x) \geq \alpha\} \text{ and } \mathrm{TVaR}_\alpha(X) = \frac{1}{1-\alpha}\int_\alpha^1 \mathrm{VaR}_t(X)dt \tag{2.12}$$

for some random variable X. From a statistical point of view, VaR is more robust than TVaR as the former is simply a quantile while the latter is the tail integral of VaR. However, from the RO perspective, TVaR is more robust than VaR since it captures more the tail risk than VaR does. Furthermore, $\mathrm{TVaR}_\alpha(X)$ is increasing in α, indicating that a larger α leads to more focus on the tail losses and thus yields more conservative decisions. In recent years, the Basel Committee on Banking Supervision confirmed the change from VaR to TVaR for market risk assessment, which signalizes the core role of TVaR for risk management in banking industry. Note that

$$\frac{1}{N}\sum_{i=1}^N (1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0))_+ = \mathbb{E}^{\hat{F}_N}[(1 - Y(\boldsymbol{\beta}^T\boldsymbol{X} + \beta_0))_+], \tag{2.13}$$

where $\hat{F}_N$ denotes the empirical joint distribution of the features and response. As such, one can re-write the objective function of (1.10) by using the empirical expectation of $(1 - Y(\boldsymbol{\beta}^T\boldsymbol{X} + \beta_0))_+$.

Since $\mathbb{E}[X] = \mathrm{TVaR}_0(X)$, an intuitive way of robusifying the classical SVM model (1.10) is replacing the empirical expectation by the empirical TVaR with $\alpha > 0$. Such replacement was investigated by Asimit et al. (2022), and the resultant SVM is named as the extreme

empirical loss (EEL) SVM. Recall from Rockafellar et al. (2000) that

$$\text{TVaR}_\alpha(X) = \text{VaR}_\alpha(X) + \frac{1}{1-\alpha}\mathbb{E}[(X - \text{VaR}_\alpha(X))_+] = \min_z \; z + \frac{1}{1-\alpha}\mathbb{E}[(X-z)_+],$$

the empirical TVaR of the losses $\{(1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0))_+\}$ is given by

$$\widehat{\text{TVaR}}_\alpha((1-Y(\boldsymbol{\beta}^T\boldsymbol{X}+\beta_0))_+) = \min_z \; z + \frac{1}{N(1-\alpha)}\sum_{i=1}^{N}\left((1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0))_+ - z\right)_+. \quad (2.14)$$

Replacing the empirical expectation with the empirical TVaR, the problem (1.10) now becomes

$$\begin{aligned}
&\min_{\boldsymbol{\beta},\beta_0} \; ||\boldsymbol{\beta}|| + \lambda\widehat{\text{TVaR}}_\alpha((1 - Y(\boldsymbol{\beta}^T\boldsymbol{X} + \beta_0))_+) \\
\implies &\min_{\boldsymbol{\beta},\beta_0,z} \; ||\boldsymbol{\beta}|| + \lambda z + \frac{\lambda}{N(1-\alpha)}\sum_{i=1}^{N}\left((1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0))_+ - z\right)_+, \\
\implies &\begin{cases}
\min\limits_{\boldsymbol{\beta},\beta_0,z,\boldsymbol{\xi}} \; ||\boldsymbol{\beta}|| + \lambda z + \dfrac{\lambda}{N(1-\alpha)}\displaystyle\sum_{i=1}^{N}\xi_i, \\
\quad \text{s.t. } \xi_i + z \geq 1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i + \beta_0) \text{ for } i = 1, 2, \ldots, N, \\
\qquad \xi_i + z \geq 0 \text{ for } i = 1, 2, \ldots, N, \\
\qquad \xi_i \geq 0 \text{ for } i = 1, 2, \ldots, N.
\end{cases}
\end{aligned} \quad (2.15)$$

The problem (2.15) is essentially a LCQP problem and can be solved by using any convex programming packages. However, the problem (2.15) now involves $3N$ inequalities as constraints, hence more computational cost is anticipated. The explicit solution is also derivable, and the interested readers can refer to the appendix of Asimit et al. (2022) for more details.

Chapter 3

Robust classifiers under Wasserstein Distance

The emergence and popularization of RO ignite the study of many decision-making problems under distributional uncertainty, and the assumption where the decision maker possesses the perfect knowledge of the distributions of either the features or responses is no longer prevailing. However, a central question needs to be answered before applying RO: how to choose the uncertainty set of distributions that can best depict the decision maker's knowledge/belief towards those distributions? As per the extant literature, there are mainly three veins for the construction of uncertainty set. The first vein is to consider a given finite set of distributions, which come from the external experts' opinions. For example, Asimit et al. (2017) and Jiang et al. (2020) studied the optimal insurance contracting problems under the finite set of loss distributions. In the second vein, the uncertainty set is constructed by assuming that only partial information of the moments of the distributions are exactly known. The recent representative works include, for example, Liu and Mao (2022) and Xie et al. (2023). The third vein applies the distance-based approach to construct the uncertainty set, in which the candidate distributions are assumed to closely surround a given benchmark distribution, where the "closeness" is measured by using a distance metric. See, for example,

Boonen and Jiang (2024) and their cited references.

In this chapter, we study a class of RO-based classifiers, where the uncertainty set of either the joint distribution of features and response or the conditional distributions of features given response is modelled via the Wasserstein ball, which encompasses all the distributions that are close to the pre-determined benchmark distribution under Wasserstein distance (WD). The main contribution of our work is to derive the tractable formulations for the RO problems. The chapter is structured as follows. Section 3.1 introduces the background of WD. Section 3.2 sets up the RO problem for the case where the uncertainty set of the joint distribution of features and response is a Wasserstein ball, and presents its tractable formulation. Section 3.3 sets up the RO problem for the case where the uncertainty set of the conditional distributions of features given response are Wasserstein balls, and presents its tractable formulation. Throughout this chapter, the following notations might be frequently used: $x \vee y = \max\{x, y\}$, $x \wedge y = \min\{x, y\}$ and $[x]_+ = \max\{0, x\}$. The indicator function $\mathbb{1}_A(x)$ equals 1 if $x \in A$ and 0 otherwise.

3.1 Wasserstein distance

Wasserstein distance is also known as the Monge-Kantorovich distance. The pth-order Wasserstein distance between two CDFs F_1 and F_2, whose supports are $\Xi_1 \subseteq \mathbb{R}^d$ and $\Xi_2 \subseteq \mathbb{R}^d$ respectively, is defined as

$$W_p(F_1, F_2) = \inf_{\substack{X \sim F_1 \\ Y \sim F_2}} \left(\mathbb{E}[||X - Y||^p]\right)^{\frac{1}{p}}, \tag{3.1}$$

where X and Y are d-dimensional random vectors. The definition (3.1) can be smoothly generalized to a more general complete and separable metric space, with $||X - Y||$ replaced by $d(X, Y)$, where $d(\cdot, \cdot)$ is a distance metric. It is easy to verify that $W_p(F_1, F_2)$ is finite if the pth moments of both X and Y exist. The optimization problem in (3.1) is typically

referred to as the Monge-Kantorovich problem, and can be re-written as

$$\inf_{\substack{X\sim F_1\\ Y\sim F_2}} (\mathbb{E}[||X-Y||^p])^{\frac{1}{p}} = \inf_{K\in\Gamma(F_1,F_2)} \left(\int_{\Xi_1\times\Xi_2} ||x-y||^p K(dx,dy)\right)^{\frac{1}{p}}, \tag{3.2}$$

where $\Gamma(F_1, F_2)$ is the set of joint distributions of X and Y, whose marginal distributions are F_1 and F_2. As such, the Wasserstein distance (3.1) is characterized by the worst-case joint distribution of X and Y, or equivalently, the worst-case copula that couples the marginal distributions F_1 and F_2 (Nelsen, 2006). A popular physical interpretation for (3.1) or (3.2) arises from the transportation theory. The quantity $||x-y||^p$ is the cost/effort for moving one unit of mass from x to y and $K(dx, dy)$ quantifies the amount of mass ought to be moved from the region around x to the region around y[1]. Hence, the Wasserstein distance quantifies the minimal cost/effort that reconfigure F_1 into F_2. More historical and recent advances in optimal transportation theory can be found in Villani et al. (2009).

Straightforwardly, if both F_1 and F_2 are degenerate distributions at x and y, then $W_p(F_1, F_2)$ reduces to the L^p distance between x and y. Furthermore, the convergence of any sequence of random variables under the Wasserstein distance is equivalent to the convergence in distribution. For more details, we refer the interested readers to Panaretos and Zemel (2019).

Unfortunately, the Wasserstein distance (3.1) generally does not possess closed-form. Some exceptional cases that allow explicit forms for the Wasserstein distance are the case where X and Y are both one-dimensional random variables and the case where the marginal distributions F_1 and F_2 are notmal. For the one-dimensional case, the Wasserstein distance admits the following representation:

$$W_p(F_1,F_2) = \left(\int_0^1 |F_1^{-1}(t) - F_2^{-1}(t)|^p dt\right)^{\frac{1}{p}}, \tag{3.3}$$

[1]Here $\int_{\Xi_2} K(dx,dy)dy = K(dx)$ is the total amount of mass moved out from the region around x, while $\int_{\Xi_1} K(dx,dy)dx = K(dy)$ is the total amount of mass moved to the region around y.

where F_1^{-1} and F_2^{-1} are quantile functions (the left-inverse of the CDF) of F_1 and F_2 respectively. Recall the L^p distance between the functions F and G

$$L_p(F, G) = \left(\int_{\mathbb{R}} |F(x) - G(x)|^p dx \right)^{\frac{1}{p}}, \tag{3.4}$$

the Wasserstein distance for the one-dimensional case (3.3) is simply the L^p distance between the quantile functions of F_1 and F_2. If $L_p(F_1, F_2) \geq 1$, Boonen and Jiang (2024) showed that

$$L_p(F_1, F_2) \leq W_p(F_1, F_2).$$

Thus, generally the L^p distance between F_1 and F_2 is not equal to the Wasserstein distance between these two CDFs. The only exceptional case is when $p = 1$, and it always holds that

$$L_1(F_1, F_2) = W_1(F_1, F_2).$$

Based on the definitions, applying either the Wasserstein distance or the L^p distance allows the CDFs F_1 and F_2 to have different supports, which is more flexible than, for example, the Kullback–Leibler divergence, where the absolute continuity condition is always required. The theoretical results established in the following two sections are under the Wasserstein distance only. Without any doubt, these results can also be extended to the cases where the L^p distance is used.

3.2 Uncertain joint distribution of features and response

3.2.1 Problem setup

As reviewed in Chapters 1 and 2, a wide class of regression and classification problems can be formulated mathematically as

$$\inf_{\boldsymbol{\beta}} \; C \cdot \sum_{i=1}^{N} L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) + R(\boldsymbol{\beta}), \tag{3.5}$$

where L is a loss function that measures the discrepancy between y_i and $f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)$, $f_{\boldsymbol{\beta}}(\boldsymbol{x}_i) = \langle \boldsymbol{\beta}, \boldsymbol{x}_i \rangle = \boldsymbol{\beta}^T \boldsymbol{x}_i$ is named as the regression function in a regression model and discriminant function in a classification model, and $R(\boldsymbol{\beta})$ is a regularization term. For instance, in a regularized linear regression model the quadratic loss function $L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) = (y_i - f_{\boldsymbol{\beta}}(\boldsymbol{x}_i))^2$ is applied, and the L^2 and L^1 norms of $\boldsymbol{\beta}$ are used as $R(\boldsymbol{\beta})$ in the ridge and lasso regression models respectively. For a binary classification problem, the binomial deviance loss function $L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) = \log[1 + e^{-y_i f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)}]$ is applied in the logistic regression model, and the hinge loss function $L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) = [1 - y_i f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)]_+$ is applied in the support vector machine (SVM). The weighting parameter C in (3.5) measures the relative importance of the aggregate loss as compared with the regularization.

The problem (3.5) can be rewritten as

$$\inf_{\boldsymbol{\beta}} \; \hat{C} \cdot \frac{1}{N} \sum_{i=1}^{N} L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) + R(\boldsymbol{\beta}), \tag{3.6}$$

with $\hat{C} := C \cdot N$, then the average loss $\frac{1}{N}\sum_{i=1}^{N} L(y_i, f_{\boldsymbol{\beta}}(\boldsymbol{x}_i)) = \mathbb{E}^{\hat{F}_N}[L(Y, f_{\boldsymbol{\beta}}(\boldsymbol{X}))]$, where $\hat{F}_N$ is the empirical distribution constructed using the samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$. In practice, the small sample size can make the use of $\hat{F}_N$ questionable as the empirical distribution often fails to capture all the characteristics of the true underlying distribution F. As such, the regression or discriminant function that is generated by solving the problem (3.6) cannot be

well generalized to other data points in population.

To address the issue, we consider a robustified version of (3.6) by employing the "min-max" approach that has been prevalent in robust optimization for years. This turns the problem (3.6) into

$$\inf_{\boldsymbol{\beta}} \hat{C}\left\{\sup_{F\in\mathcal{B}(\hat{F}_N,\epsilon)} \mathbb{E}^F[L(Y, f_{\boldsymbol{\beta}}(\boldsymbol{X}))]\right\} + R(\boldsymbol{\beta}), \tag{3.7}$$

where the set $\mathcal{B}(\hat{F}_N, \epsilon)$ is an ϵ-uncertainty ball that centers at the benchmark distribution, which is the empirical distribution $\hat{F}_N$ in our current setup. The benchmark distribution could be replaced by others based on the external expert's opinions. In this paper, within the uncertainty ball the Wasserstein metric is applied to measure the distance between the candidate distribution and the benchmark distribution. Let F_1 and F_2 be two probability distributions supported on Ξ, and $\Gamma(F_1, F_2)$ denote the collection of joint probability distributions with marginals F_1 and F_2, we adopt a generalized Wasserstein distance in the sequel

$$W(F_1, F_2) = \inf_{K\in\Gamma(F_1,F_2)} \int_{\Xi\times\Xi} d(\xi_1, \xi_2) K(d\xi_1, d\xi_2), \tag{3.8}$$

where $d(\cdot, \cdot)$ is a distance metric on Ξ. Under the Wasserstein distance, the uncertainty ball $\mathcal{B}(\hat{F}_N, \epsilon)$ is also named as the Wasserstein ball, and is represented as

$$\mathcal{B}(\hat{F}_N, \epsilon) = \left\{F : W(F, \hat{F}_N) \leq \epsilon)\right\}, \tag{3.9}$$

where ϵ is a pre-set radius that captures the ambiguity aversion of the decision maker. If $\epsilon = 0$, then $\mathcal{B}(\hat{F}_N, \epsilon)$ reduces to a singleton set. To avoid the trivial case, we assume in the remaining of this paper that $\epsilon > 0$.

For the choice of the distance metric $d(\cdot, \cdot)$, we follow the literature, e.g. Lee and Mehrotra (2015) and Shafieezadeh Abadeh et al. (2015), and assume that

$$d((\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2)) = ||\boldsymbol{x}_1 - \boldsymbol{x}_2|| + \frac{\delta}{2}||y_1 - y_2||, \tag{3.10}$$

where $||\cdot||$ is the L^2 norm, and $\delta \geq 0$ measures the relative importance of the distance between the response variables as compared to the distance between the feature vectors.

For the choice of the loss function L, we restrict ourselves to those for the classification problems, with focus on the binary case, where

$$L(Y, f_{\boldsymbol{\beta}}(\boldsymbol{X})) = L(Y f_{\boldsymbol{\beta}}(\boldsymbol{X})),$$

with $Y \in \{-1, 1\}$. Here a correct prediction of Y is achieved if the sign of $f_{\boldsymbol{\beta}}(\boldsymbol{X})$ agrees with that of Y. Hence, an appropriate loss function should penalize the observations with negative $Y f_{\boldsymbol{\beta}}(\boldsymbol{X})$. To name but a few, some choices for the function L are listed below:

- Binomial deviance: $L(t) = \log[1 + e^{-t}]$, which results in the logistic regression.
- Hinge loss: $L(t) = [1 - t]_+$, which results in the classical soft-margin SVM.
- Truncated hinge loss: $L(t) = [1 - t]_+ \wedge a$ with $a \geq 1$, which results in a soft-margin SVM that is insensitive to observations with extremely negative $Y f_{\boldsymbol{\beta}}(\boldsymbol{x})$ (Wu and Liu, 2007).
- Pinball loss: $L(t) = [a(1 - t)] \vee (1 - t)$ with $a \leq 0$ (Huang et al., 2013).
- "Huberised" square hinge loss: $L(t) = -4t\mathbb{1}_{(-\infty,-1)}(t) + [1 - t]_+^2 \mathbb{1}_{[-1,\infty)}(t)$ (Rosset and Zhu, 2007).

A desirable feature for the loss function is classification calibration or Fisher consistency (Bartlett et al., 2006). A loss function is said to be classification calibrated if

$$\begin{aligned} f_{bayes}(\boldsymbol{x}) &= \arg\min_f \ \mathbb{E}[L(Y f(\boldsymbol{x}))] \\ &= \begin{cases} 1, & \mathbb{P}(Y = 1|\boldsymbol{x}) > \mathbb{P}(Y = -1|\boldsymbol{x}), \\ -1, & \mathbb{P}(Y = 1|\boldsymbol{x}) < \mathbb{P}(Y = -1|\boldsymbol{x}). \end{cases} \end{aligned}$$

We refer the interested readers to Asimit et al. (2022) for a sufficient condition for a loss function being classification calibrated.

Throughout this paper, we assume the following for the loss function.

Assumption 1. *The function $L(t)$ is a proper convex function on $\mathbb{R}$. The lower and upper bounds of its subdifferential are a and b, i.e. $\underline{\partial} L(t) \in [a, b]$, for all $t \in \mathbb{R}$.*

It is easy to calculate that, the subdifferential for the binomial deviance loss function is bounded by 0 and 1; the subdifferential for the hinge loss function is bounded by -1 and 0; the subdifferential for the Pinball loss function is bounded by -1 and $-a$; and the subdifferential for the "huberised" square hinge loss function is bounded by -4 and 0. As such, most aforementioned loss functions satisfy Assumption 1, among which the hinge loss and Pinball loss functions are classification calibrated.

The next section develops a tractable formulation for the problem of (3.7).

3.2.2 Tractable formulation

We tackle the inner problem of (3.7) first, which is the most challenging part. For that purpose, we need the following minimax theorem.

Theorem 3.1 (Minimax theorem of Fan (1953))**.** *Let Ξ_1 be a non-empty compact convex Hausdorff topological vector space and Ξ_2 be a convex set. If $\mathcal{H}$ is a real-valued function defined on $\Xi_1 \times \Xi_2$ and satisfies*

- *$\xi_1 \mapsto \mathcal{H}(\xi_1, \xi_2)$ is convex and lower-semi-continuous on Ξ_1 for each $\xi_2 \in \Xi_2$;*
- *$\xi_2 \mapsto \mathcal{H}(\xi_1, \xi_2)$ is concave on Ξ_2 for each $\xi_1 \in \Xi_1$,*

then,

$$\inf_{\xi_1 \in \Xi_1} \sup_{\xi_2 \in \Xi_2} \mathcal{H}(\xi_1, \xi_2) = \sup_{\xi_2 \in \Xi_2} \inf_{\xi_1 \in \Xi_1} \mathcal{H}(\xi_1, \xi_2).$$

The following lemma will play a pivotal role in simplifying the problem (3.7).

Lemma 3.2. *If the function l satisfies Assumption 1, then*

$$\sup_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ l(\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda||\hat{\boldsymbol{x}} - \boldsymbol{x}|| \right\} = \begin{cases} l(\boldsymbol{\beta}^T\hat{\boldsymbol{x}}), & \text{if } ||\boldsymbol{\beta}||_* \leq \dfrac{\lambda}{|a| \vee |b|}, \\ +\infty, & \text{otherwise}, \end{cases} \tag{3.11}$$

where $\lambda > 0$, and $||\cdot||_*$ is the dual norm of $||\cdot||$.

Proof. Since the objective function $l(\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda||\hat{\boldsymbol{x}} - \boldsymbol{x}||$ is the difference between two convex functions, the first-order condition cannot be directly applied. We adopt here the envelope technique as applied in Shafieezadeh Abadeh et al. (2015). Note that the function $\hat{l}(t) = \eta t - l(t)$ is concave, which is differentiable almost everywhere. The derivative $\hat{l}'(t) = \eta - l'(t)$ is decreasing over $\mathbb{R}$, whose roots are $[\underline{t}, \overline{t}]$, where

$$\underline{t} = \inf\left\{ t \in \mathbb{R} : \hat{l}'(t) \leq 0 \right\}, \quad \overline{t} = \sup\left\{ t \in \mathbb{R} : \hat{l}'(t) \geq 0 \right\}.$$

By convention, $\underline{t} = +\infty$ if $\left\{ t \in \mathbb{R} : \hat{l}'(t) \leq 0 \right\} = \emptyset$, while $\overline{t} = -\infty$ if $\left\{ t \in \mathbb{R} : \hat{l}'(t) \geq 0 \right\} = \emptyset$. Hence, the conjugate function (Boyd and Vandenberghe, 2004) of $l(t)$ is given by

$$l^*(\eta) = \sup_{t\in\mathbb{R}} \{\eta t - l(t)\} = \begin{cases} < +\infty, & \text{if } \eta \in [a, b], \\ +\infty, & \text{if otherwise.} \end{cases}$$

This leads the conjugate function of $h_{\boldsymbol{\beta}}(\boldsymbol{x}) := l(\boldsymbol{\beta}^T\boldsymbol{x})$ to be

$$h^*_{\boldsymbol{\beta}}(\xi) = \sup_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \xi^T\boldsymbol{x} - h_{\boldsymbol{\beta}}(\boldsymbol{x}) \right\} = \begin{cases} l^*(\eta), & \text{if } \xi = \eta\boldsymbol{\beta} \text{ for some } \eta \in [a, b], \\ +\infty, & \text{otherwise.} \end{cases}$$

Since the function $h_{\boldsymbol{\beta}}(\boldsymbol{x})$ is a proper convex function in $\boldsymbol{x}$, we have

$$h_{\boldsymbol{\beta}}(\boldsymbol{x}) = h^{**}_{\boldsymbol{\beta}}(\boldsymbol{x}) = \sup_{\xi\in\mathbb{R}^n} \left\{ \boldsymbol{x}^T\xi - h^*_{\boldsymbol{\beta}}(\xi) \right\} = \sup_{\eta\in[a,b]} \left\{ \eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta) \right\}. \tag{3.12}$$

To this step, the function $h_{\boldsymbol{\beta}}(\boldsymbol{x})$ has been written as the upper envelope of the linear functions $\left\{\eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta)\right\}_{\eta\in[a,b]}$.

Recall the definition of the dual norm $||\boldsymbol{q}||_* = \sup_{\boldsymbol{x}} \frac{\boldsymbol{q}^T\boldsymbol{x}}{||\boldsymbol{x}||}$, one has

$$\lambda||x|| = \sup_{||\boldsymbol{q}||_*\leq\lambda} \boldsymbol{q}^T\boldsymbol{x}.$$

As such,

$$\begin{aligned}
\sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{h_{\boldsymbol{\beta}}(\boldsymbol{x}) - \lambda||\hat{\boldsymbol{x}}-\boldsymbol{x}||\right\} &= \sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{h^{**}_{\boldsymbol{\beta}}(\boldsymbol{x}) - \lambda||\hat{\boldsymbol{x}}-\boldsymbol{x}||\right\} \\
&= \sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{\sup_{\eta\in[a,b]}\left(\eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta)\right) - \lambda||\hat{\boldsymbol{x}}-\boldsymbol{x}||\right\} \\
&= \sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{\sup_{\eta\in[a,b]}\left(\eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta)\right) - \sup_{||\boldsymbol{q}||_*\leq\lambda}\boldsymbol{q}^T(\hat{\boldsymbol{x}}-\boldsymbol{x})\right\} \\
&= \sup_{\boldsymbol{x}\in\mathbb{R}^n}\sup_{\eta\in[a,b]}\inf_{||\boldsymbol{q}||_*\leq\lambda}\left\{\eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T(\hat{\boldsymbol{x}}-\boldsymbol{x})\right\} \\
&= \sup_{\eta\in[a,b]}\sup_{\boldsymbol{x}\in\mathbb{R}^n}\inf_{||\boldsymbol{q}||_*\leq\lambda}\left\{\eta\boldsymbol{\beta}^T\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T(\hat{\boldsymbol{x}}-\boldsymbol{x})\right\} \\
&= \sup_{\eta\in[a,b]}\sup_{\boldsymbol{x}\in\mathbb{R}^n}\inf_{||\boldsymbol{q}||_*\leq\lambda}\left\{(\eta\boldsymbol{\beta}^T + \boldsymbol{q}^T)\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T\hat{\boldsymbol{x}}\right\}
\end{aligned}$$

Note that the set $\{\boldsymbol{q} : ||\boldsymbol{q}||_* \leq \lambda\}$ is convex and compact and $\mathbb{R}^n$ is convex. Furthermore, the objective function $(\eta\boldsymbol{\beta}^T + \boldsymbol{q}^T)\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T\hat{\boldsymbol{x}}$ is linear in both $\boldsymbol{x}$ and $\boldsymbol{q}$. Thus, Theorem 3.1 is applicable here, which yields

$$\begin{aligned}
&\sup_{\eta\in[a,b]}\sup_{\boldsymbol{x}\in\mathbb{R}^n}\inf_{||\boldsymbol{q}||_*\leq\lambda}\left\{(\eta\boldsymbol{\beta}^T + \boldsymbol{q}^T)\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T\hat{\boldsymbol{x}}\right\} \\
=\ &\sup_{\eta\in[a,b]}\inf_{||\boldsymbol{q}||_*\leq\lambda}\sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{(\eta\boldsymbol{\beta}^T + \boldsymbol{q}^T)\boldsymbol{x} - l^*(\eta) - \boldsymbol{q}^T\hat{\boldsymbol{x}}\right\}. \qquad (3.13)
\end{aligned}$$

For the inner supremum problem, it is easy to find that

$$\sup_{\boldsymbol{x}\in\mathbb{R}^n}\left\{(\eta\boldsymbol{\beta}^T+\boldsymbol{q}^T)\boldsymbol{x}-l^*(\eta)-\boldsymbol{q}^T\hat{\boldsymbol{x}}\right\}=\begin{cases}-l^*(\eta)-\boldsymbol{q}^T\hat{\boldsymbol{x}}, & \text{if } \eta\boldsymbol{\beta}^T+\boldsymbol{q}^T=0,\\ +\infty, & \text{otherwise.}\end{cases}$$

For the non-trivial case, we have $\boldsymbol{q}^T=-\eta\boldsymbol{\beta}^T$, and thus

$$||\boldsymbol{q}||_*\le\lambda\Longrightarrow\ ||\eta\boldsymbol{\beta}^T||_*\le\lambda\Longrightarrow\ |\eta|||\boldsymbol{\beta}^T||_*\le\lambda\Longrightarrow\ ||\boldsymbol{\beta}^T||_*\le\frac{\lambda}{|a|\vee|b|}.$$

This leads (3.13) to be

$$\begin{cases}\sup\limits_{\eta\in[a,b]}\left\{-l^*(\eta)+\eta\boldsymbol{\beta}^T\hat{\boldsymbol{x}}\right\}, & \text{if } ||\boldsymbol{\beta}^T||_*\le\dfrac{\lambda}{|a|\vee|b|},\\ +\infty, & \text{otherwise,}\end{cases}$$

which, as per (3.12), yields (3.11). The proof is complete. □

With the help of Lemma 3.2, we are ready to present the CP-formulation of the inner problem of (3.7) for binary classification.

Theorem 3.3. *Let Assumption 1 hold. If the Wasserstein ball $\mathcal{B}(\hat{F}_N,\epsilon)$ is equipped with the distance metric (3.10), then*

$$\begin{aligned}&\sup_{F\in\mathcal{B}(\hat{F}_N,\epsilon)}\mathbb{E}^F[L(Y(\boldsymbol{\beta}^TX+\beta_0))]\\ =&\inf_{\lambda\ge(|a|\vee|b|)||\boldsymbol{\beta}||}\lambda\epsilon+\frac{1}{N}\sum_{i=1}^N\left\{L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))\vee\left(L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta\right)\right\}.\end{aligned}\tag{3.14}$$

Proof. For the ease of presentation, we let $\xi=(\boldsymbol{x},y)$, $\Xi=\mathcal{X}\times\{-1,1\}$, where $\mathcal{X}$ is the support of X, and $h_{\boldsymbol{\beta}}(\xi)=l(y\boldsymbol{\beta}^T\boldsymbol{x})=L(y(\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0))$. The problem on the left-hand side

of (3.14) is now

$$\sup_{F} \int_{\Xi} h_{\boldsymbol{\beta}}(\xi) F(d\xi)$$

The empirical distribution $\hat{F}_N$ is a discrete distribution built on the sample points $\{\xi_i\}_{i=1}^N$, where $\xi_i = (\boldsymbol{x}_i, y_i)$ and all the points are assigned the equal probabilities. Then finding the worst-case distribution within $\mathcal{B}(\hat{F}_N, \epsilon)$ is equivalent to finding the worst-case joint distribution K that satisfies

$$\int_{\Xi} \sum_{i=1}^{N} d(\xi, \xi_i) K(d\xi, \xi_i) \leq \epsilon, \quad \sum_{i=1}^{N} K(d\xi, \xi_i) = F(d\xi) \text{ and } \int_{\Xi} K(d\xi, \xi_i) = \frac{1}{N} \text{ for } i \in \{1, \ldots, N\}. \tag{3.15}$$

Thus, we have

$$\begin{aligned}
\int_{\Xi} h_{\boldsymbol{\beta}}(\xi) F(d\xi) &= \int_{\Xi} h_{\boldsymbol{\beta}}(\xi) \sum_{i=1}^{N} K(d\xi, \xi_i) \\
&= \sum_{i=1}^{N} \int_{\Xi} h_{\boldsymbol{\beta}}(\xi) K(d\xi, \xi_i) \\
&= \sum_{i=1}^{N} \int_{\Xi} h_{\boldsymbol{\beta}}(\xi) K(d\xi | \xi_i) \mathbb{P}(\xi_i) \\
&= \frac{1}{N} \sum_{i=1}^{N} \int_{\Xi} h_{\boldsymbol{\beta}}(\xi) K^i(d\xi),
\end{aligned}$$

where K^i denote the conditional distribution of ξ given the observation ξ_i. Similarly, we have

$$\int_{\Xi} \sum_{i=1}^{N} d(\xi, \xi_i) K(d\xi, \xi_i) = \frac{1}{N} \sum_{i=1}^{N} \int_{\Xi} d(\xi, \xi_i) K^i(d\xi).$$

To this step, the left-side of (3.14) can be written as

$$\begin{aligned} &\sup_{K^i} \ \frac{1}{N}\sum_{i=1}^{N}\int_{\Xi} h_{\boldsymbol{\beta}}(\xi)K^i(d\xi), \\ &\text{s.t.} \ \frac{1}{N}\sum_{i=1}^{N}\int_{\Xi} d(\xi,\xi_i)K^i(d\xi) \leq \epsilon. \end{aligned} \tag{3.16}$$

It is apparent that both the objective function and the constraint of (3.16) is linear in $(K^1,\ldots,K^N)$, thus the Slater's condition holds (Boyd and Vandenberghe, 2004). Due to the strong duality, the problem (3.16) is turned into

$$\begin{aligned} &\sup_{K^i}\inf_{\lambda\geq 0} \ \frac{1}{N}\sum_{i=1}^{N}\int_{\Xi} h_{\boldsymbol{\beta}}(\xi)K^i(d\xi) - \lambda\left(\frac{1}{N}\sum_{i=1}^{N}\int_{\Xi} d(\xi,\xi_i)K^i(d\xi) - \epsilon\right) \\ =&\inf_{\lambda\geq 0} \ \lambda\epsilon + \left\{\sup_{K^i} \ \frac{1}{N}\sum_{i=1}^{N}\int_{\Xi}\left(h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i)\right)K^i(d\xi)\right\}. \end{aligned} \tag{3.17}$$

For the inner problem of (3.17), it is straightforward that

$$\begin{aligned} \sup_{K^i} \ \frac{1}{N}\sum_{i=1}^{N}\int_{\Xi}\left(h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i)\right)K^i(d\xi) \leq& \frac{1}{N}\sum_{i=1}^{N}\sup_{\xi\in\Xi}\left(h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i)\right)\int_{\Xi}K^i(d\xi) \\ =&\frac{1}{N}\sum_{i=1}^{N}\sup_{\xi\in\Xi}\left(h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i)\right), \end{aligned}$$

where the equality is attained when K^i assigns probability one to $\arg\max_{\xi\in\Xi}\left(h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i)\right)$. Moreover, note that

$$\sup_{\xi\in\Xi} \ h_{\boldsymbol{\beta}}(\xi) - \lambda d(\xi,\xi_i) = \sup_{y\in\{-1,1\}}\sup_{\boldsymbol{x}\in\boldsymbol{X}}\left\{l(y\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda d((\boldsymbol{x},y),(\boldsymbol{x}_i,y_i))\right\}. \tag{3.18}$$

For the inner problem of (3.18), we have the following two cases.

- If $y = 1$, then

$$\begin{aligned}
&\sup_{\boldsymbol{x}\in\boldsymbol{X}} \; l(\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda d((\boldsymbol{x}, 1), (\boldsymbol{x}_i, y_i)) \\
=&\sup_{\boldsymbol{x}\in\boldsymbol{X}} \; l(\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda||\boldsymbol{x} - \boldsymbol{x}_i|| - \frac{\lambda\delta}{2}||1 - y_i|| \\
=&\begin{cases} l(\boldsymbol{\beta}^T\boldsymbol{x}_i) - \dfrac{\lambda\delta}{2}||1 - y_i||, & \text{if } \lambda \geq ||\boldsymbol{\beta}||_* \left(|a| \vee |b|\right), \\ +\infty, & \text{otherwise,} \end{cases}
\end{aligned}$$

  where the last equality is due to Lemma 3.2.

- If $y = -1$, then similarly

$$\begin{aligned}
&\sup_{\boldsymbol{x}\in\boldsymbol{X}} \; l(-\boldsymbol{\beta}^T\boldsymbol{x}) - \lambda d((\boldsymbol{x}, -1), (\boldsymbol{x}_i, y_i)) \\
=&\begin{cases} l(-\boldsymbol{\beta}^T\boldsymbol{x}_i) - \dfrac{\lambda\delta}{2}||1 + y_i||, & \text{if } \lambda \geq ||\boldsymbol{\beta}||_* \left(|a| \vee |b|\right), \\ +\infty, & \text{otherwise.} \end{cases}
\end{aligned}$$

We further note that

- If $y_i = 1$, then

$$l(\boldsymbol{\beta}^T\boldsymbol{x}_i) - \frac{\lambda\delta}{2}||1 - y_i|| = l(y_i\boldsymbol{\beta}^T\boldsymbol{x}_i)$$

  and

$$l(-\boldsymbol{\beta}^T\boldsymbol{x}_i) - \frac{\lambda\delta}{2}||1 + y_i|| = l(-y_i\boldsymbol{\beta}^T\boldsymbol{x}_i) - \lambda\delta.$$

- If $y_i = -1$, then

$$l(\boldsymbol{\beta}^T\boldsymbol{x}_i) - \frac{\lambda\delta}{2}||1 - y_i|| = l(-y_i\boldsymbol{\beta}^T\boldsymbol{x}_i) - \lambda\delta$$

  and

$$l(-\boldsymbol{\beta}^T\boldsymbol{x}_i) - \frac{\lambda\delta}{2}||1 + y_i|| = l(y_i\boldsymbol{\beta}^T\boldsymbol{x}_i).$$

Therefore, (3.18) becomes

$$
\begin{aligned}
&\sup_{y\in\{-1,1\}}\sup_{\boldsymbol{x}\in\boldsymbol{X}}\left\{l(y\boldsymbol{\beta}^T\boldsymbol{x})-\lambda d((\boldsymbol{x},y),(\boldsymbol{x}_i,y_i))\right\}\\
=&\begin{cases}\max\left\{l(\boldsymbol{\beta}^T\boldsymbol{x}_i)-\dfrac{\lambda\delta}{2}||1-y_i||, l(-\boldsymbol{\beta}^T\boldsymbol{x}_i)-\dfrac{\lambda\delta}{2}||1+y_i||\right\}, & \text{if } \lambda\geq||\boldsymbol{\beta}||_*\left(|a|\vee|b|\right),\\ +\infty, & \text{otherwise,}\end{cases}\\
=&\begin{cases}\max\left\{l(y_i\boldsymbol{\beta}^T\boldsymbol{x}_i), l(-y_i\boldsymbol{\beta}^T\boldsymbol{x}_i)-\lambda\delta\right\}, & \text{if } \lambda\geq||\boldsymbol{\beta}||_*\left(|a|\vee|b|\right),\\ +\infty, & \text{otherwise.}\end{cases}
\end{aligned}
$$

This, together with (3.17) and $||\cdot||_*=||\cdot||$[2], yields (3.14). The proof is complete. □

We define the following set

$$\mathcal{A}:=\left\{(\boldsymbol{\beta},\lambda):\boldsymbol{\beta}\in\mathbb{R}^n,\ \lambda\geq||\boldsymbol{\beta}||(|a|\vee|b|)\right\}. \tag{3.19}$$

A tractable formulation for (3.7) is thus given by

$$\inf_{(\boldsymbol{\beta},\lambda)\in\mathcal{A}}\hat{C}\lambda\epsilon+\frac{\hat{C}}{N}\sum_{i=1}^{N}\left\{L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))\vee\left(L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta\right)\right\}+R(\boldsymbol{\beta}). \tag{3.20}$$

The objective function of (3.20) can also be written as

$$
\begin{aligned}
&\hat{C}\lambda\epsilon+\frac{\hat{C}}{N}\sum_{i=1}^{N}\left\{L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))\vee\left(L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta\right)\right\}+R(\boldsymbol{\beta})\\
=&\frac{\hat{C}}{N}\sum_{i=1}^{N}\left\{L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))\vee\left(L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta\right)+\lambda\epsilon\right\}+R(\boldsymbol{\beta})\\
=&\frac{\hat{C}}{N}\sum_{i=1}^{N}\tilde{L}(y_i(\boldsymbol{\beta}^Tx_i+\beta_0))+R(\boldsymbol{\beta}),
\end{aligned}
\tag{3.21}
$$

where $\tilde{L}(x):=L(x)\vee(L(-x)-\lambda\delta)+\lambda\epsilon$ can be understood as the induced loss function under the Wasserstein uncertainty. Here, $L(-x)-\lambda\delta$ is the loss generated by the possible

[2]The dual norm of L^2 norm is still L^2 norm.

mislabelled response, and $\lambda\epsilon$ is the loss generated by the overall uncertainty. It is well known that the maximum of a set of convex functions is still convex, thus $\tilde{L}$ preserves the convexity of L. In an extreme case where δ is sufficiently large such that $L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta \leq L(y_i(\boldsymbol{\beta}^T x_i+\beta_0))$ for all $i\in\{1,\ldots,N\}$, (3.21) simplifies to

$$\frac{\hat{C}}{N}\sum_{i=1}^{N}\tilde{L}(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))+R(\boldsymbol{\beta})=\frac{\hat{C}}{N}\sum_{i=1}^{N}(L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))+\lambda\epsilon)+R(\boldsymbol{\beta}),$$

which reduces (3.20) to

$$\begin{aligned}
&\inf_{(\boldsymbol{\beta},\lambda)\in\mathcal{A}}\hat{C}\lambda\epsilon+\frac{\hat{C}}{N}\sum_{i=1}^{N}\left\{L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))\vee\left(L(-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))-\lambda\delta\right)\right\}+R(\boldsymbol{\beta})\\
=&\inf_{(\boldsymbol{\beta},\lambda)\in\mathcal{A}}\frac{\hat{C}}{N}\sum_{i=1}^{N}(L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))+\lambda\epsilon)+R(\boldsymbol{\beta})\\
=&\inf_{\boldsymbol{\beta}\in\mathbb{R}^n}\frac{\hat{C}}{N}\sum_{i=1}^{N}L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))+\hat{C}||\boldsymbol{\beta}||(|a|\vee|b|)\epsilon+R(\boldsymbol{\beta})\\
=&\inf_{\boldsymbol{\beta}\in\mathbb{R}^n}\frac{\hat{C}}{N}\sum_{i=1}^{N}L(y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))+\tilde{R}(\boldsymbol{\beta}),
\end{aligned}\tag{3.22}$$

where $\tilde{R}(\boldsymbol{\beta}):=\hat{C}||\boldsymbol{\beta}||(|a|\vee|b|)\epsilon+R(\boldsymbol{\beta})$ is understood as a new penalty term on the regression coefficients. This turns the problem (3.20) into the same form of (3.6). The equation (3.22) enjoys a very intuitive interpretation: mislabeling the response variable is not allowed if δ in (3.10) is extremely large but the Wasserstein ball $\mathcal{B}(\hat{F}_N)$ is small.

We end this section by remarking that: though the problem (3.20) is of the CP-form, optimizing the parameters $\boldsymbol{\beta}$ and λ simultaneously may still be computational costly. With that being said, more tractable forms are available for specific yet commonly used loss functions. We leave that to the later sections.

3.3 Uncertain conditional distributions of features

As shown in Section 3.2.1, the distribution of $(\boldsymbol{x}, y)$ is perturbed when taking the ambiguity into account. In practice, it is more often the case that only the distribution of features gets perturbed. Note that for a binary classification problem, (3.6) can be re-written as

$$\inf_{\boldsymbol{\beta}} \frac{C_1}{m_1} \sum_{y_i=1} L(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) + \frac{C_2}{m_2} \sum_{y_i=-1} L(-(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)) + R(\boldsymbol{\beta}), \tag{3.23}$$

where $m_1 = \#\{i : y_i = 1\}$, $m_2 = \#\{i : y_i = -1\}$, and $C_1 = Cm_1$ and $C_2 = Cm_2$. The above optimization problem admits the following expectation representation:

$$\inf_{\boldsymbol{\beta}} C_1 \mathbb{E}^{\hat{F}_{\boldsymbol{x}|y=1}}[L(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0)] + C_2 \mathbb{E}^{\hat{F}_{\boldsymbol{x}|y=-1}}[L(-(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0))] + R(\boldsymbol{\beta}), \tag{3.24}$$

where $\hat{F}_{\boldsymbol{x}|y=1}$ and $\hat{F}_{\boldsymbol{x}|y=-1}$ are the two empirical conditional distributions of $\boldsymbol{x}$. As such, alternative to perturbing the joint distribution of $(\boldsymbol{x}, y)$, one can perturb the conditional distributions $\hat{F}_{\boldsymbol{x}|y=1}$ and $\hat{F}_{\boldsymbol{x}|y=-1}$ and re-formulate the problem (3.24) as

$$\inf_{\boldsymbol{\beta}} C_1 \left\{ \sup_{F_1 \in \mathcal{B}(\hat{F}_{\boldsymbol{x}|y=1}, \epsilon_1)} \mathbb{E}^{F_1}[L(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0)] \right\} + C_2 \left\{ \sup_{F_2 \in \mathcal{B}(\hat{F}_{\boldsymbol{x}|y=-1}, \epsilon_2)} \mathbb{E}^{F_2}[L(-(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0))] \right\} + R(\boldsymbol{\beta}), \tag{3.25}$$

where $\mathcal{B}(\hat{F}_{\boldsymbol{x}|y=1}, \epsilon_1)$ and $\mathcal{B}(\hat{F}_{\boldsymbol{x}|y=-1}, \epsilon_2)$ are two Wasserstein balls that center at $\hat{F}_{\boldsymbol{x}|y=1}$ and $\hat{F}_{\boldsymbol{x}|y=-1}$ with radiuses $\epsilon_1 > 0$ and $\epsilon_2 > 0$ respectively.

Similar as Section 3.2.2, we seek the tractable formulation of the problem (3.25). Analogous to Theorem 3.3, the two inner problems of (3.25) have nice tractable forms:

$$\begin{aligned} \sup_{F_1 \in \mathcal{B}(\hat{F}_{\boldsymbol{x}|y=1}, \epsilon_1)} \mathbb{E}^{F_1}[L(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0)] &= \inf_{\lambda_1 \geq (|a| \vee |b|)||\boldsymbol{\beta}||} \lambda_1 \epsilon_1 + \frac{1}{m_1} \sum_{y_i=1} L(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0), \\ &= (|a| \vee |b|)||\boldsymbol{\beta}||\epsilon_1 + \frac{1}{m_1} \sum_{y_i=1} L(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0), \end{aligned}$$

and

$$\sup_{F_2 \in \mathcal{B}(\hat{F}_{\boldsymbol{x}|y=-1}, \epsilon_2)} \mathbb{E}^{F_2}[L(-(\boldsymbol{\beta}^T \boldsymbol{x} + \beta_0))] = \inf_{\lambda_2 \geq (|a| \vee |b|)||\boldsymbol{\beta}||} \lambda_2 \epsilon_2 + \frac{1}{m_2} \sum_{y_i=-1} L(-(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0))$$
$$= (|a| \vee |b|)||\boldsymbol{\beta}||\epsilon_2 + \frac{1}{m_2} \sum_{y_i=-1} L(-(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)).$$

This turns the problem (3.25) into

$$\inf_{\boldsymbol{\beta} \in \mathbb{R}^n} (|a| \vee |b|)||\boldsymbol{\beta}||(C_1\epsilon_1 + C_2\epsilon_2) + \frac{C_1}{m_1} \sum_{y_i=1} L(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0) + \frac{C_2}{m_2} \sum_{y_i=-1} L(-(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)) + R(\boldsymbol{\beta}),$$

or equivalently,

$$\inf_{\boldsymbol{\beta} \in \mathbb{R}^n} C \sum_{i=1}^{N} L(y_i(\boldsymbol{\beta}^T \boldsymbol{x}_i + \beta_0)) + (|a| \vee |b|)||\boldsymbol{\beta}||(C_1\epsilon_1 + C_2\epsilon_2) + R(\boldsymbol{\beta}). \tag{3.26}$$

We remark that: if $\epsilon_1 = \epsilon_2 = \epsilon$, then

$$C_1\epsilon_1 + C_2\epsilon_2 = Cm_1\epsilon + Cm_2\epsilon = CN\epsilon = \hat{C}\epsilon,$$

which reduces the problem (3.26) to (3.22). Therefore, perturbing the empirical conditional distributions with equal-radius Wasserstein balls is tantamount to perturbing the joint distribution of the features and response with an extremely large δ in the distance metric (3.10).

3.4 WD-SVM

In this section, we investigate the robust SVM under the Wasserstein distance, where the penalty term is the L^2 norm of $\boldsymbol{\beta}$. In such a case, the subdifferential of the loss function is bounded by 0 and 1. Under the hinge loss function and the L^2-norm penality, the problem

(3.20) becomes

$$\inf_{(\boldsymbol{\beta},\lambda)\in\mathcal{A}} \hat{C}\lambda\epsilon + \frac{\hat{C}}{N}\sum_{i=1}^{N}\left\{(1-y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))_+ \vee ((1+y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0))_+ - \lambda\delta)\right\} + ||\boldsymbol{\beta}||. \quad (3.27)$$

Note that $(1-y\boldsymbol{\beta}^T\boldsymbol{x})_+ \vee ((1+y\boldsymbol{\beta}^T\boldsymbol{x})_+ - \lambda\delta)$ is natually the solution to the following problem

$$\begin{cases} \inf\limits_{\xi\in\mathbb{R}} \xi, \\ \text{s.t. } \xi \geq 1 - y(\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0), \\ \qquad \xi \geq 1 + y(\boldsymbol{\beta}^T\boldsymbol{x}+\beta_0) - \lambda\delta, \\ \qquad \xi \geq 0. \end{cases} \quad (3.28)$$

Therefore, the problem (3.27) can be written as

$$\begin{cases} \inf\limits_{\substack{\boldsymbol{\beta},\boldsymbol{\xi}\in\mathbb{R}^n \\ \beta_0,\lambda\in\mathbb{R}}} \hat{C}\lambda\epsilon + \dfrac{\hat{C}}{N}\sum\limits_{i=1}^{N}\xi_i + ||\boldsymbol{\beta}||, \\ \quad \text{s.t. } \xi_i \geq 1 - y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0) \text{ for } i = 1,2,\ldots,N, \\ \qquad \xi_i \geq 1 + y_i(\boldsymbol{\beta}^T\boldsymbol{x}_i+\beta_0) - \lambda\delta \text{ for } i = 1,2,\ldots,N, \\ \qquad \xi_i \geq 0 \text{ for } i = 1,2,\ldots,N, \\ \qquad \lambda \geq ||\boldsymbol{\beta}||. \end{cases} \quad (3.29)$$

Hence, the problem (3.27) is essentially a linearly constrained second-order-cone programming problem, which differs from the problems (2.10) and (2.15), which are linearly constrained quadratic programming problems.

Under the same setting, the problem (3.26) becomes

$$\inf_{\boldsymbol{\beta}\in\mathbb{R}^n,\beta_0\in\mathbb{R}} C\sum_{i=1}^{N}(1-y_i(\boldsymbol{\beta}\boldsymbol{x}_i+\beta_0))_+ + (C_1\epsilon_1 + C_2\epsilon_2 + 1)||\boldsymbol{\beta}||, \quad (3.30)$$

which shares the same form of (2.10). As such, for SVM the consideration of the feature un-

certainty via Wasserstein balls is equivalent to the consideration of the box-type uncertainty as described in Section (2.2.1).

Both of the problems (3.27) and (3.30) can be solved by using the readily available packages in most computing software. We close this section by remarking the selection of the radius of Wasserstein ball. Generally, the selection of ϵ is subjective, reflecting the ambiguity aversion (or the level of conservativeness) of the decision maker. Typically, a more ambiguity-averse decision maker applies a smaller ϵ in her model. Instead of subjectively selecting ϵ, statistical manners are also available in the literature. For example, De Wet (2002) derived the limiting distribution for a class of goodness-of-fit test statistics based on the weighted second-order Wasserstein distance. The confidence interval of such limiting distribution can be utilized for choosing the appropriate ϵ. This is not further investigated in this thesis.

Chapter 4

Simulation analysis

In this chapter we apply the WD-SVM to the synthetic data with and without contamination. The sensitivity analysis is conducted by varying the radius of the Wasserstein ball. The comparison with SVM models under the hinge loss and TVaR is also carried out to show the advantages of our WD-SVM. Note that the box-type uncertainty is as same as the Wasserstein-type uncertainty due to the reasoning presented in Section 3.4, which is thus excluded from the comparison. We do not compare the statistically robustified models with the RO robustified models, as these two robustification mechanisms adopt completely different notions, making the comparison lack of rationale. Section 4.1 focuses on the simulated data without contamination. Section 4.2 focuses on the simulated data with contamination.

4.1 Synthetic data without contamination

In this section, the data are generated from a known distribution, similarly as in Asimit et al. (2022). More specifically, we first generate the response variable Y, which is a Bernoulli random variable with the probability of success $p = 0.5$. Then, conditioning on the value of Y, the vector of features $\boldsymbol{X}$ is assumed to follow the bivariate normal distributions:

$$\boldsymbol{X}|Y = 1 \sim N(\mu, \Sigma), \quad \boldsymbol{X}|Y = -1 \sim N(-\mu, \Sigma), \tag{4.1}$$

where the mean vector and the variance-covariance matrix are given by

$$\mu = [0.5, 3]^T, \quad \Sigma = \begin{bmatrix} 0.2 & 0 \\ 0 & 3 \end{bmatrix}. \tag{4.2}$$

Our experiment comprises 500 simulation rounds, and in each round we sample 100, 500, and 5000 points, with 10% of these points serving for the training purpose, while the rest of points are used for testing. The average correct classification rate (CCR) is defined as the correct classification ratios averaged through the 500 simulation rounds. We show the sensitivity of average CCR with respect to the radius of Wasserstein ball in Figure 4.1, where different colors correspond to different sample sizes. For the choice of ϵ, we use the same ones as adopted in Shafieezadeh Abadeh et al. (2015), i.e. $\epsilon \in \{10^{-5}, 10^{-4}, \ldots, 10^{-1}, 1\}$. It is not surprising that larger sample size leads to higher average CCR, since the case considered here is perfectly separable. More interestingly, for the ϵ ranging from 10^{-5} to 10^{-1}, the average CCR almost does not change and can attain $94\% \sim 97\%$ level. However, the average CCR drops dramatically when ϵ changes from 10^{-1} to 1, and the size of drawdown increases with respect to the sample size. The intuition behind this observation is that: given a larger sample size, the empirical joint distribution of the features and response is getting closer to the true underlying distribution (which is also supported by Glivenko–Cantelli theorem), and using larger ϵ leads to further deviation from the true underlying distribution, which hence lowers the average CCR.

We also compare the average CCR of WD-SVM with those under the classical SVM and the EEL-SVM. Table 4.1 summarizes the average CCRs for the different sample sizes under the three SVM models. Note that for EEL-SVM, fewer extremal observations are penalized when α becomes larger, and the classical SVM is a special case of the EEL-SVM with $\alpha = 0$. Table 4.1 indicates that the EEL-SVM produces very stable average CCR for α ranging from 0 to at least 0.9 for this perfectly separable case. However, when α becomes extremely large (e.g., when $\alpha = 0.99$), the EEL-SVM shows lower average CCR as very

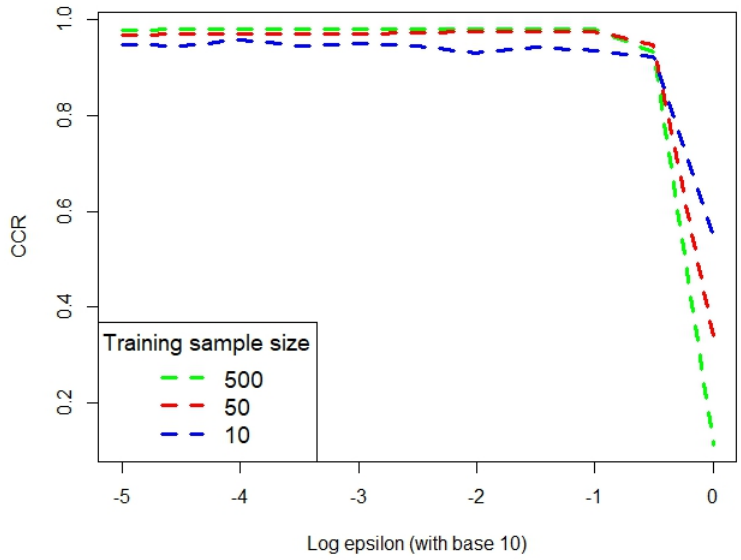


Figure 4.1: The CCR of WD-SVM on non-contaminated synthetic data.

few observations are penalized in the model. The WD-SVM, however, is comparable to the classical SVM, and even slightly outperforms the latter for some chosen ϵ. Thus, without data contamination, perturbing the joint distribution of features and response can still be beneficiary to the decision maker in terms of improving the classification rate.

Sample size	Classical SVM	EEL-SVM ($\alpha = 0.9$)	EEL-SVM ($\alpha = 0.99$)	WD-SVM ($\epsilon = 10^{-3}$)	WD-SVM ($\epsilon = 10^{-1}$)
100	94.31%	94.20%	94.00%	95.16%	93.58%
500	96.91%	96.02%	94.51%	97.02%	97.66%
5000	97.92%	97.92%	93.89%	97.96%	98.04%

Table 4.1: Comparison of the average CCRs under different SVMs.

Though comparing the average CCR is often of practical interest, the hyperplane that produces the same average CCR is not unique for most cases, and SVM is only one of such hyperplanes that creates the largest margin between the classes. For our case, where the feature vectors follow the conditional bivariate normal distributions with the same variance-covariance matrix, the Bayesian decision boundary can be explicitly derived (see Section 4.4

of James et al. (2013)), which is

$$x^T\Sigma^{-1}\mu - \frac{1}{2}\mu^T\Sigma^{-1}\mu = x^T\Sigma^{-1}(-\mu) - \frac{1}{2}(-\mu)^T\Sigma^{-1}(-\mu) \Longrightarrow x_2 = 2.5x_1.$$

Note that the hyperplanes of SVMs can be written as $x_2 = m_1 x_1 + m_0$. Thus, another fair way to evaluate the different SVMs is to compare the distance (Asimit et al., 2022)

$$D = |\hat{m}_1 - 2.5|\hat{\sigma}_{m_1} + |\hat{m}_0 - 0|\hat{\sigma}_{m_0}, \tag{4.3}$$

where $\hat{m}_1$ and $\hat{m}_0$ denote respectively the estimated m_1 and m_0, and $\hat{\sigma}_{m_1}$ and $\hat{\sigma}_{m_0}$ are the estimated standard deviation of $\hat{m}_1$ and $\hat{m}_0$. The distance metric (4.3) uses a mean-standard-deviation criterion to assess the performance of SVM, with the reference separation hyperplane determined by the Bayesian decision boundary.

Sample size	Classical SVM	EEL-SVM ($\alpha = 0.9$)	EEL-SVM ($\alpha = 0.99$)	WD-SVM ($\epsilon = 10^{-3}$)	WD-SVM ($\epsilon = 10^{-1}$)
100	1.981	2.132	2.520	1.268	0.587
500	0.253	0.546	0.997	0.155	0.207
5000	0.048	0.071	0.870	0.017	0.012

Table 4.2: Comparison of the distance D under different SVMs.

The calculated distances are summarized in Table 4.2. As expected, when the sample size becomes larger, the distance is smaller, indicating that the resulting hyperplane gets closer to the Bayesian decision boundary. If comparing the distances for the classical SVM and EEL-SVM, it is found that the distance is increasing with α, implying that penalizing fewer observations in the SVM model makes the separation hyperplane further deviate from the Bayesian decision boundary. Interestingly, the distances calculated under the WD-SVM are smaller than those calculated under the classical SVM or EEL-SVM, which shows from another perspective the advantage of incorporating the Wasserstein-type uncertainty into the SVM models.

4.2 Synthetic data with contamination

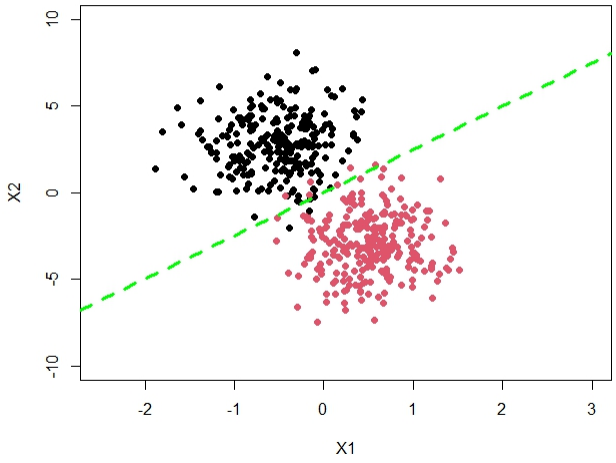
4.2.1 Original feature space

In this section, we investigate the performance of the classical SVM, EEL-SVM and WD-SVM on synthetic data in the presence of contamination. Following Asimit et al. (2022), we assume that part of the sample data are subject to external contamination, where the original data points are replaced with the new ones, whose feature vectors are generated around some given focal point. We assume that the focal point is $(2, -4)$, and the responses are labelled with 1 or -1 with equal probabilities. The variance-covariance matrix for the new data points is

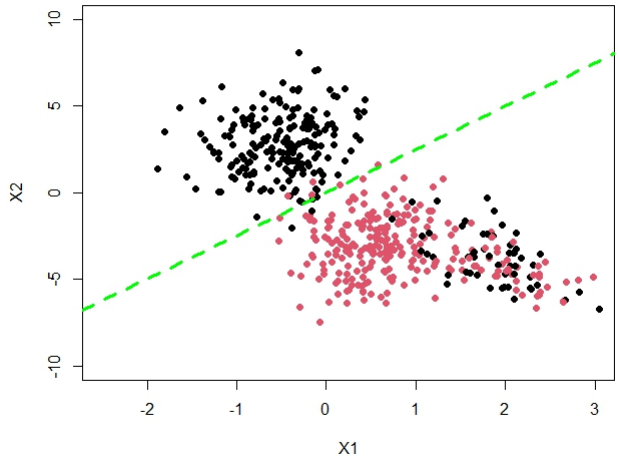
$$\begin{bmatrix} 0.2 & -0.5 \\ -0.5 & 3 \end{bmatrix}.$$

For the contamination rate – the proportion of the original data being replaced with the new ones, we choose 20%, 30% and 40% to see their effect on the average CCRs of different SVM models. Figure 4.2 illustrates the data contamination based on different contamination rates, where the green dashed line refers to the Bayesian decision boundary in the absence of data contamination.

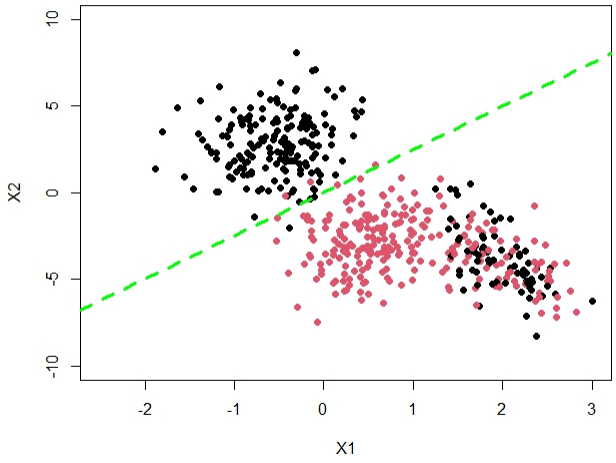
Under different contamination rates and sample sizes, we apply the three kinds of SVM models to the synthetic data. The average CCRs are shown in Table 4.3. Apparently, the average CCR of EEL-SVM is greatly impacted by the data contamination since EEL-SVM concerns the extremal points, whose existence would significantly bias the classification results. Nevertheless, when the contamination rate changes, it is found that the average CCR of EEL-SVM does not change as significantly as others, showing its robustness under the extremal perturbation. In the presence of data contamination, the average CCR of WD-SVM is the highest for most cases. However, when contamination rate changes, its average CCR also changes dramatically. These experiments deliver some straightforward yet important messages regarding the selection of the radius of Wasserstein ball: selecting a small ϵ makes



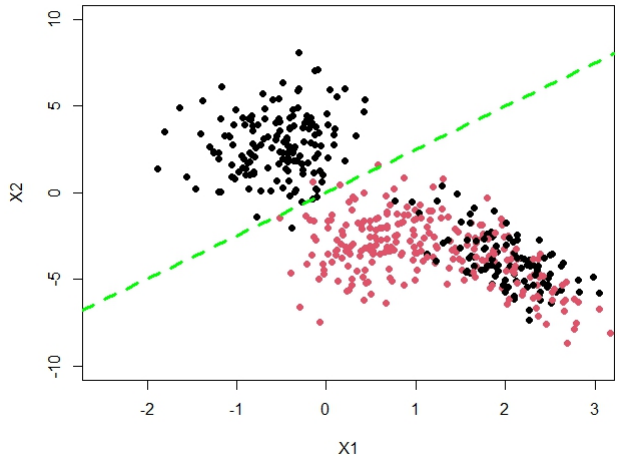
(a) Data without contamination.



(b) Data with contamination (20% rate).



(c) Data with contamination (30% rate).



(d) Data with contamination (40% rate).

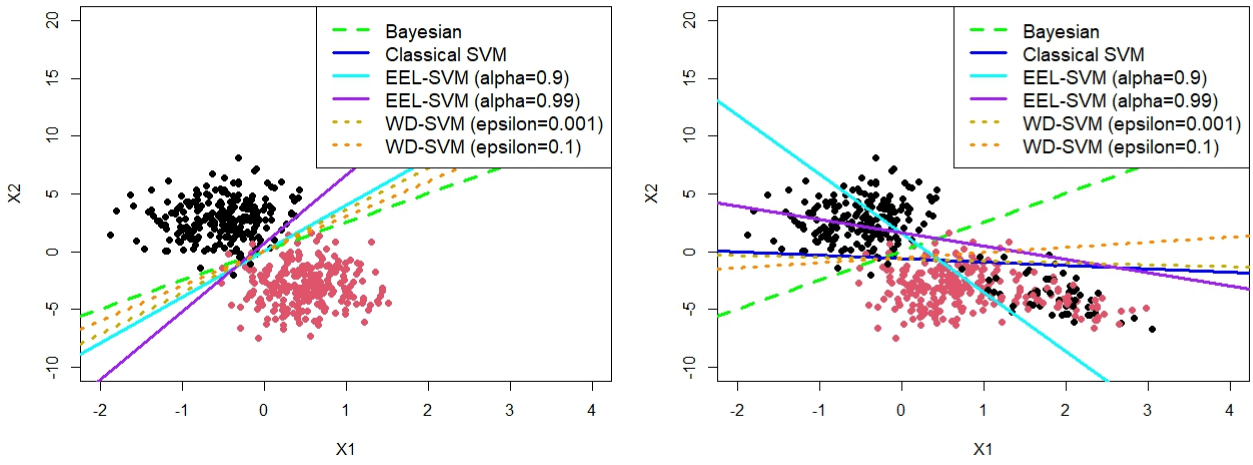
Figure 4.2: The comparison between the non-contaminated and contaminated data.

the other candidate distributions "closely" surround the benchmark one (i.e., the empirical distribution) and thus the improvement of average CCR is minor and the resulting model can still be sensitive to the extremal points. However, selecting a large ϵ might overweight the outlier points, which would bias the classification results similarly as EEL-SVM.

Besides comparing the average CCRs of different SVM models, we also visualize the separating hyperplanes in Figure 4.3. In the absence of data contamination, Figure 4.3 (a) shows

Sample size	Contamination rate	Classical SVM	EEL-SVM ($\alpha = 0.9$)	EEL-SVM ($\alpha = 0.99$)	WD-SVM ($\epsilon = 10^{-3}$)	WD-SVM ($\epsilon = 10^{-1}$)
100	20%	78.37%	70.69%	70.71%	79.09%	79.63%
	30%	73.36%	63.42%	65.86%	71.84%	72.42%
	40%	67.90%	55.12%	56.48%	68.49%	66.49%
500	20%	84.02%	59.89%	62.25%	84.82%	86.23%
	30%	78.05%	55.82%	59.63%	78.29%	79.57%
	40%	72.84%	52.09%	53.75%	73.14%	73.75%
5000	20%	85.37%	62.20%	71.38%	85.44%	86.79%
	30%	79.70%	55.42%	64.53%	79.79%	81.52%
	40%	74.65%	51.33%	59.92%	74.82%	76.46%

Table 4.3: Comparison of the average CCRs under different SVMs in the presence of data contamination.



(a) Data without contamination.

(b) Data with contamination (20% rate).

Figure 4.3: The separating hyperplanes for non-contaminated and contaminated data.

that the separating hyperplanes resulting from WD-SVM models are closer to the Bayesian decision boundary. The separating hyperplanes derived based on the classical SVM and EEL-SVM with $\alpha = 0.9$ almost overlap each other, and the separating hyperplane resulting from the EEL-SVM with $\alpha = 0.99$ is the furthest away from the Bayesian decision boundary. These observations are consistent with the results from Table 4.2. In the presence of data contamination, as shown in Figure 4.3 (b), all the hyperplanes are more or less impacted. Notably, the hyperplane for EEL-SVM with $\alpha = 0.9$ is the furthest away from the Bayesian

decision boundary, which explains its lowest classification accuracy rate. An interesting observation is that: when α increases from 0.9 to 0.99, the EEL-SVM separating hyperplane gets closer to the Bayesian decision boundary, which explains the increased classification accuracy rate. By considering the Wasserstein-type uncertainty, the resulting hyperplane can get closer to the Bayesian decision boundary and thus improve the classification accuracy. Note that the hyperplanes plotted in Figure 4.3 are based on one round of simulation. The calculated distances D (as shown by (4.3)) based on the 500 rounds of simulation are summarized in the following table. It can be found that in most cases the WD-SVM with $\epsilon = 0.1$ has the shortest distance, which echoes its highest classification accuracy rate for most cases as shown in Table 4.3.

Sample size	Contamination rate	Classical SVM	EEL-SVM ($\alpha = 0.9$)	EEL-SVM ($\alpha = 0.99$)	WD-SVM ($\epsilon = 10^{-3}$)	WD-SVM ($\epsilon = 10^{-1}$)
100	20%	10.445	13.117	9.562	9.107	7.246
	30%	6.445	12.210	8.499	9.826	9.788
	40%	11.666	16.999	17.011	11.680	10.034
500	20%	3.526	20.696	20.478	3.215	1.819
	30%	2.968	16.387	21.743	3.296	3.388
	40%	3.473	16.425	25.480	3.519	3.529
5000	20%	0.747	19.283	15.343	1.041	0.657
	30%	1.009	16.615	12.406	1.048	0.623
	40%	1.245	20.225	19.031	1.068	0.691

Table 4.4: Comparison of the distance D under different SVMs in the presence of data contamination.

4.2.2 Transformed feature space

In this section we investigate the application of WD-SVM to the transformed feature space. The use of transformed features can enlarge the feature space, enhancing the separability of the sample points. The conventional ways of transforming the original features are via the basis expansions, such as using polynomial and splines basis functions. See, for example, James et al. (2013) and Hastie et al. (2009) for the elementary applications of the above-mentioned two basis functions in SVM models. In this section, we do not aim to provide a

comprehensive view of the effectiveness of these basis expansions, but rather to show how the implementation of various SVM models based on the transformed feature space can improve the classification accuracy. In the following, we apply the 2nd-degree polynomial basis functions to transform the original features, i.e.

$$[X_1, X_2] \implies [h_1(X_1, X_2), \ldots, h_5(X_1, X_2)], \tag{4.4}$$

where $h_1(X_1, X_2) = \sqrt{2}X_1$, $h_2(X_1, X_2) = \sqrt{2}X_2$, $h_3(X_1, X_2) = X_1^2$, $h_4(X_2, X_2) = X_2^2$ and $h_5(X_1, X_2) = \sqrt{2}X_1X_2$. The hyperplane built on the new feature space is of the form $\sum_{i=1}^{5} \beta_i h_i(X_1, X_2) + \beta_0$. We apply the different SVM models to the transformed data points $\{(h_1(\boldsymbol{x}_i), \ldots, h_5(\boldsymbol{x}_i))\}_{i=1}^{N}$ and summarize the average CCRs in Table 4.5.

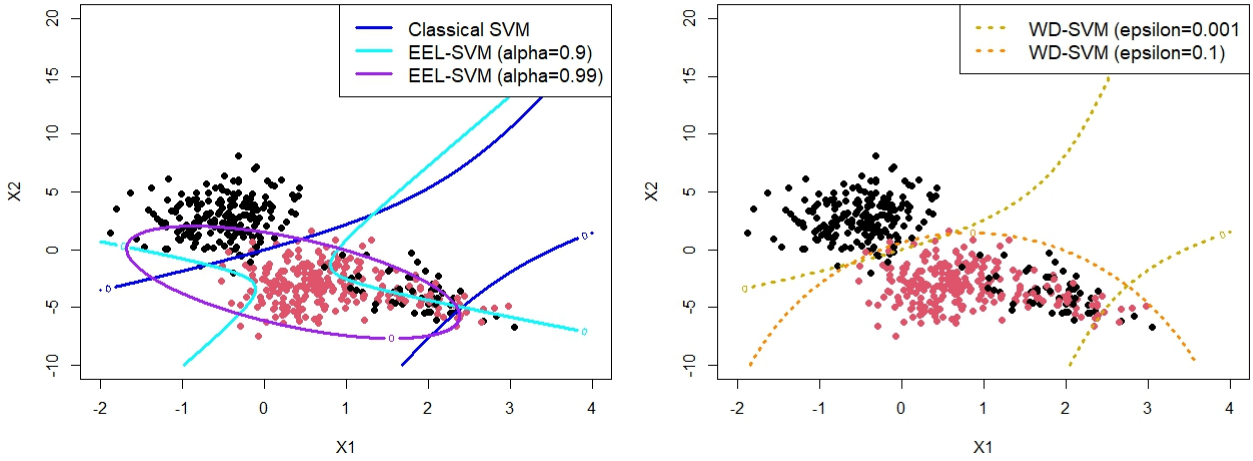
Sample size	Contamination rate	Classical SVM	EEL-SVM ($\alpha = 0.9$)	EEL-SVM ($\alpha = 0.99$)	WD-SVM ($\epsilon = 10^{-3}$)	WD-SVM ($\epsilon = 10^{-1}$)
100	20%	79.02%	79.14%	77.32%	79.76%	78.08%
	30%	72.92%	71.49%	70.90%	72.10%	71.53%
	40%	68.16%	64.96%	64.20%	67.47%	67.19%
500	20%	85.78%	75.66%	79.15%	86.48%	86.08%
	30%	81.00%	62.14%	70.70%	81.33%	81.15%
	40%	76.10%	62.58%	66.34%	75.93%	75.63%
5000	20%	88.16%	66.28%	78.19%	88.18%	87.38%
	30%	83.44%	63.53%	70.68%	83.32%	82.53%
	40%	78.51%	62.91%	67.92%	78.52%	77.49%

Table 4.5: Comparison of the average CCRs under the transformed features in the presence of data contamination.

By comparing Table 4.5 with Table 4.3, obviously the average CCRs are improved for most cases, showing the advantage of using the transformed feature space. Interestingly, the WD-SVM, especially the one with $\epsilon = 0.1$, no longer has the highest classification rate for any case, indicating the inferior choice of ϵ. This is in contrast to the conclusion achieved based on Table 4.3.

We also exhibit the resulting hyperplanes within the original feature space, which are now non-linear, in Figure 4.4. Note that the main goal of the current simulation study is to

distinguish between the main clusters of the two classes (shown as the clusters of black and red points on the left-lower corner of Figure 4.4), and this is achieved by the classical SVM and WD-SVM models.



(a) Classical SVM and EEL-SVM.

(b) WD-SVM.

Figure 4.4: The polynomial separating hyperplanes for contaminated data (20% contamination rate).

We close this section by remarking that other basis functions, such as higher-degree polynomial or radial basis functions, are also popular choices for transforming the original feature space. The SVM models in this chapter can be applied similarly within those transformed feature spaces. The comparison of the classification accuracy rates by using different basis functions is not further pursued in this thesis.

Chapter 5

Conclusion and future research

The robustness of classification models is of longstanding concern, and there are vast literature on robustification of the existing models. This thesis proposes a robust optimization perspective to robustify a class of binary classifiers, where the expected loss is maximized by considering the worst case distribution that belongs to a pre-determined uncertainty set. The uncertainty set being used in this thesis is a Wasserstein ball, which encompasses all the distributions that surround a benchmark distribution (e.g., the empirical distribution of the sample points) within a certain distance, where the distance is measured by Wasserstein metric. The tractable form of the optimization problem is derived, and solving the corresponding SVM model turns out to be a second-order cone programming problem. The connection between the SVM models under the Wasserstein-type uncertainty and the box-type uncertainty is also investigated, where an equivalence can be achieved if the Wasserstein-type uncertainties are only applied to the conditional distributions of feature vectors. Unlike the classical SVM model, the developed Wasserstein-distance-based SVM models do not admit closed-form solutions for the parameters in general. Nonetheless, the second-order cone programming problem can be solved by using the readily available packages in most modern optimization and statistics software, which shows the applicability of our proposed model for most practical problems. The numerical examples show that the WD-SVM can outperform

the classical SVM under many circumstances, regardless of the presence or absence of data contamination.

The study carried out in this thesis paves the way for several future research opportunities.

(a). As shown in the numerical part, the selection of radius of the Wasserstein ball can significantly affect the classification accuracy. A small radius pushes all the candidate distributions to the benchmark distribution, which makes the model still sensitive to the extremal outlier points. However, a large radius may overweight the extremal outlier points, which may sacrifice the classification accuracy for the stability. The balancing point between these two aspects deserves more future research.

(b). This thesis only focuses on the tractable forms and the computational aspects for the cases where the joint distribution of features and response and the conditional distributions of features are subject to uncertainty. The closed-form or analytical solutions for the coefficients of separating hyperplanes might exist if only single point perturbation is considered (see also Asimit et al. (2022)), which allow for more investigation of the analytical properties of robustified classifiers under Wasserstein distance.

(c). A natural technical extension of the current study is robustifying multicategory classifiers under Wasserstein distance. Such problem can be tackled via solving a sequence of binary classification problems, or similarly as Liu and Shen (2006) where all the classes are treated simultaneously. Doubtlessly, such extension would bring non-trivial challenges, and we leave such study to future research.

(d). Last but not least, the developed results in Chapter 2 can be applied to other classification models, as long as Assumption 1 is satisfied. We leave the implementation of other robustified classifiers to the future real data analyses.

Bibliography

Alexandru V Asimit, Valeria Bignozzi, Ka Chun Cheung, Junlei Hu, and Eun-Seok Kim. Robust and pareto optimality of insurance contracts. *European Journal of Operational Research*, 262(2):720–732, 2017.

Alexandru V Asimit, Ioannis Kyriakou, Simone Santoni, Salvatore Scognamiglio, and Rui Zhu. Robust classification via support vector machines. *Risks*, 10(8):154, 2022.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Jose Blanchet, Lin Chen, and Xun Yu Zhou. Distributionally robust mean-variance portfolio selection with wasserstein distances. *Management Science*, 68(9):6382–6410, 2022.

Tim J Boonen and Wenjun Jiang. Robust insurance design with distortion risk measures. *European Journal of Operational Research*, 316(2):694–706, 2024.

Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

J Paul Brooks. Support vector machines with the ramp loss and the hard margin loss. *Operations research*, 59(2):467–479, 2011.

T De Wet. Goodnes-of-fit tests for location and scale families based on a weighted l 2-wasserstein distance measure. *Test*, 11:89–107, 2002.

Ky Fan. Minimax theorems. *Proceedings of the National Academy of Sciences*, 39(1):42–47, 1953.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Xiaolin Huang, Lei Shi, and Johan AK Suykens. Support vector machine classifier with pinball loss. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):984–997, 2013.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

Wenjun Jiang, Marcos Escobar-Anel, and Jiandong Ren. Optimal insurance contracts under distortion risk measures with ambiguity aversion. *ASTIN Bulletin: The Journal of the IAA*, 50(2):619–646, 2020.

Changhyeok Lee and Sanjay Mehrotra. A distributionally-robust approach for finding support vector machines. *Available from Optimization Online*, 2015.

Haiyan Liu and Tiantian Mao. Distributionally robust reinsurance with value-at-risk and conditional value-at-risk. *Insurance: Mathematics and Economics*, 107:393–417, 2022.

Yufeng Liu and Xiaotong Shen. Multicategory ψ-learning. *Journal of the American Statistical Association*, 101(474):500–509, 2006.

Yufeng Liu, Xiaotong Shen, and Hani Doss. Multicategory ψ-learning and support vector machine: computational tools. *Journal of Computational and Graphical Statistics*, 14(1): 219–236, 2005.

Roger B Nelsen. *An introduction to copulas*. Springer, 2006.

Victor M Panaretos and Yoav Zemel. Statistical aspects of wasserstein distances. *Annual review of statistics and its application*, 6:405–431, 2019.

R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

Saharon Rosset and Ji Zhu. Piecewise linear regularized solution paths. *The Annals of Statistics*, pages 1012–1030, 2007.

Soroosh Shafieezadeh Abadeh, Peyman M Mohajerin Esfahani, and Daniel Kuhn. Distributionally robust logistic regression. *Advances in Neural Information Processing Systems*, 28, 2015.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Yichao Wu and Yufeng Liu. Robust truncated hinge loss support vector machines. *Journal of the American Statistical Association*, 102(479):974–983, 2007.

Yichao Wu and Yufeng Liu. Adaptively weighted large margin classifiers. *Journal of Computational and Graphical Statistics*, 22(2):416–432, 2013.

Xinqiao Xie, Haiyan Liu, Tiantian Mao, and Xiao Bai Zhu. Distributionally robust reinsurance with expectile. *ASTIN Bulletin: The Journal of the IAA*, 53(1):129–148, 2023.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of machine learning research*, 10(7), 2009.

Appendix A

Code for simulation

A.1 For synthetic data without contamination

```
library(MASS)
library(CVXR)
library(quadprog)
library(demogR)
library(Matrix)

# WD-SVM
epsilon_s<-c(-3,-1)
ACCR_WD<-rep(0,length(epsilon_s))

for (j in 1:length(epsilon_s)) {
set.seed(2024)
CCR_WD<-rep(0,100)
for (k in 1:100) {
  mu0<-c(0.5,-3)
```

```
Sigma0<-diag(c(0.2,3))

Ntotal<-5000
N<-0.1*Ntotal  # training data size
B<-rbinom(Ntotal,1,0.5)
B0<-(B==1)-(B==0)

X0<-matrix(0,length(B0),2)
for (i in 1:length(B0)) {
  X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
}

Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2])
ind_t<-sample(1:Ntotal, N, replace = F)
Data0<-Data[ind_t,]  # training set
Data1<-Data[-ind_t,]  # testing set

C=100
epsilon=10^epsilon_s[j]

dVEC<-matrix(c(rep(0,3),rep(C,N),C*N*epsilon),nrow=1)
AMAT<-cbind(matrix(c(Data0$Y*Data0$X1,Data0$Y*Data0$X2,Data0$Y),nrow=N,byrow=F),
diag(rep(1,N)),matrix(rep(0,N),ncol=1))
AMAT<-rbind(AMAT,cbind(matrix(c(-Data0$Y*Data0$X1,-Data0$Y*Data0$X2,-Data0$Y),
nrow=N,byrow=F),diag(rep(1,N)),matrix(rep(1,N),ncol=1)))
AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),3),matrix(rep(0,N+3),ncol=1)))
AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
```

```
  bv<-matrix(c(rep(1,N),rep(1,N),rep(0,N)))

  beta<-Variable(N+4)
  objective<-Minimize(1/2*(beta[1]^2+beta[2]^2)+dVEC%*%beta)
  constraint1<-AMAT%*%beta>=bv
  constraint2<-beta[N+4]>=norm(beta[1:2],type="2")
  problem<-Problem(objective, constraints = list(constraint1, constraint2))
  result_WD<-solve(problem)
  beta_hat<-result_WD$getValue(beta)[1:3]
  lambda_hat<-result_WD$getValue(beta)[N+4]

  y_pred<-ifelse(as.matrix(Data1[,c(2,3)])%*%beta_hat[1:2]+beta_hat[3]>0,1,-1)
  CCR_WD[k]<-mean(y_pred==Data1$Y)
}
ACCR_WD[j]<-mean(CCR_WD)
}

ACCR_WD1<-ACCR_WD
ACCR_WD2<-ACCR_WD
ACCR_WD3<-ACCR_WD

plot(epsilon_s, ACCR_WD3, type="l", lty=2, lwd=3, col="green",
xlab="Log epsilon (with base 10)", ylab="CCR")
lines(epsilon_s, ACCR_WD2, lty=2, lwd=3, col="red")
lines(epsilon_s, ACCR_WD1, lty=2, lwd=3, col="blue")
legend("bottomleft", title="Training sample size", legend=c("500", "50", "10"),
col=c("green", "red", "blue"), lty=c(2,2,2), lwd=c(3,3,3), cex=1.2)
```

```
# EEL-SVM
alpha_s=c(0,0.9,0.99)
ACCR_EEL<-rep(0,length(alpha_s))

for (j in 1:length(alpha_s)) {
set.seed(2024)
CCR_EEL<-rep(0,100)
for (k in 1:100) {
  mu0<-c(0.5,-3)
  Sigma0<-diag(c(0.2,3))

  Ntotal<-100
  N<-0.1*Ntotal  # training data size
  B<-rbinom(Ntotal,1,0.5)
  B0<-(B==1)-(B==0)

  X0<-matrix(0,length(B0),2)
  for (i in 1:length(B0)) {
    X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
  }

  Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2])
  ind_t<-sample(1:Ntotal, N, replace = F)
  Data0<-Data[ind_t,]  # training set
  Data1<-Data[-ind_t,]  # testing set

  C=100
```

```
  alpha=alpha_s[j]
  AMAT<-rbind(cbind(matrix(c(Data0$Y*Data0$X1,Data0$Y*Data0$X2,Data0$Y),
  nrow=N,byrow=F), diag(rep(1,N)),matrix(rep(1,N),ncol=1)),
  cbind(odiag(rep(1,N),3),matrix(rep(1,N+3),ncol=1)))
  AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
  AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),3),matrix(rep(0,N+3),ncol=1)))
  AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
  bv<-matrix(c(rep(1,N),rep(0,2*N)),ncol=1)

  dVEC<--C*N*matrix(c(rep(0,3),rep(1/(N*(1-alpha)),N),1),nrow=1)
  DMAT<-diag(c(rep(1,2),rep(0,N+2)))
  res2_EEL<-solve.QP(Dmat=as.matrix(nearPD(DMAT)$mat), dvec=dVEC,
  Amat=t(AMAT), bvec=bv)
  beta_hat<-res2_EEL$solution[1:3]

  y_pred<-ifelse(as.matrix(Data1[,c(2,3)])%*%beta_hat[1:2]+beta_hat[3]>0,1,-1)
  CCR_EEL[k]<-mean(y_pred==Data1$Y)
}
ACCR_EEL[j]<-mean(CCR_EEL)
}

ACCR_EEL1<-ACCR_EEL
ACCR_EEL2<-ACCR_EEL
ACCR_EEL3<-ACCR_EEL
```

A.2 For synthetic data with contamination

A.2.1 Original feature space

```
# WD-SVM
epsilon_s<-c(-3,-1)
ACCR_WD<-rep(0,length(epsilon_s))
dbayes_WD<-rep(0,length(epsilon_s))

for (j in 1:length(epsilon_s)) {
  set.seed(2024)
  CCR_WD<-rep(0,100)
  for (k in 1:100) {
    mu0<-c(0.5,-3)
    Sigma0<-diag(c(0.2,3))

    Ntotal<-5000
    N<-0.1*Ntotal  # training data size
    B<-rbinom(Ntotal,1,0.5)
    B0<-(B==1)-(B==0)

    X0<-matrix(0,length(B0),2)
    for (i in 1:length(B0)) {
      X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
    }

    Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2])
```

```
Calpha=0.4 # proportion of contaminated data
Ncon<-Calpha*Ntotal
B_con<-rbinom(Ncon, 1, 0.5)
B0_con<-(B_con==1)-(B_con==0)
X0_con<-matrix(0,length(B0_con),2)
mu0_con<-c(2,-4)
Sigma0_con<-matrix(c(0.2,-0.5,-0.5,3),nrow=2,byrow=T)
for (i in 1:length(B0_con)) {
  X0_con[i,]<-mvrnorm(1,mu0_con,Sigma0_con)
}
Data_con<-data.frame(Y=B0_con,X1=X0_con[,1],X2=X0_con[,2])
Data<-rbind(Data[1:((1-Calpha)*Ntotal),],Data_con)

ind_t<-sample(1:Ntotal, N, replace = F)
Data0<-Data[ind_t,]  # training set
Data1<-Data[-ind_t,]  # testing set

C=100
epsilon=10^epsilon_s[j]

dVEC<-matrix(c(rep(0,3),rep(C,N),C*N*epsilon),nrow=1)
AMAT<-cbind(matrix(c(Data0$Y*Data0$X1,Data0$Y*Data0$X2,Data0$Y),
nrow=N,byrow=F),diag(rep(1,N)),matrix(rep(0,N),ncol=1))
AMAT<-rbind(AMAT,cbind(matrix(c(-Data0$Y*Data0$X1,-Data0$Y*Data0$X2,-Data0$Y),
nrow=N,byrow=F),diag(rep(1,N)),matrix(rep(1,N),ncol=1)))
AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),3),matrix(rep(0,N+3),ncol=1)))
AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
```

```
    bv<-matrix(c(rep(1,N),rep(1,N),rep(0,N)))

    beta<-Variable(N+4)
    objective<-Minimize(1/2*(beta[1]^2+beta[2]^2)+dVEC%*%beta)
    constraint1<-AMAT%*%beta>=bv
    constraint2<-beta[N+4]>=norm(beta[1:2],type="2")
    problem<-Problem(objective, constraints = list(constraint1, constraint2))
    result_WD<-solve(problem)
    beta_hat<-result_WD$getValue(beta)[1:3]
    lambda_hat<-result_WD$getValue(beta)[N+4]

    y_pred<-ifelse(as.matrix(Data1[,c(2,3)])%*%beta_hat[1:2]+beta_hat[3]>0,1,-1)
    CCR_WD[k]<-mean(y_pred==Data1$Y)
  }
  ACCR_WD[j]<-mean(CCR_WD)
}

ACCR_WD1<-ACCR_WD
ACCR_WD2<-ACCR_WD
ACCR_WD3<-ACCR_WD

# EEL-SVM
alpha_s=c(0,0.9,0.99)
ACCR_EEL<-rep(0,length(alpha_s))
dbayes_EEL<-rep(0,length(alpha_s))

for (j in 1:length(alpha_s)) {
  set.seed(2024)
```

```
CCR_EEL<-rep(0,100)
m1<-rep(0,100)
m0<-rep(0,100)
for (k in 1:100) {
  mu0<-c(0.5,-3)
  Sigma0<-diag(c(0.2,3))

  Ntotal<-5000
  N<-0.1*Ntotal  # training data size
  B<-rbinom(Ntotal,1,0.5)
  B0<-(B==1)-(B==0)

  X0<-matrix(0,length(B0),2)
  for (i in 1:length(B0)) {
    X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
  }

  Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2])

  Calpha=0.2 # proportion of contaminated data
  Ncon<-Calpha*Ntotal
  B_con<-rbinom(Ncon, 1, 0.5)
  B0_con<-(B_con==1)-(B_con==0)
  X0_con<-matrix(0,length(B0_con),2)
  mu0_con<-c(2,-4)
  Sigma0_con<-matrix(c(0.2,-0.5,-0.5,3),nrow=2,byrow=T)
  for (i in 1:length(B0_con)) {
```

```
  X0_con[i,]<-mvrnorm(1,mu0_con,Sigma0_con)
}
Data_con<-data.frame(Y=B0_con,X1=X0_con[,1],X2=X0_con[,2])
Data<-rbind(Data[1:((1-Calpha)*Ntotal),],Data_con)

ind_t<-sample(1:Ntotal, N, replace = F)
Data0<-Data[ind_t,]  # training set
Data1<-Data[-ind_t,]  # testing set

C=100

alpha=alpha_s[j]
AMAT<-rbind(cbind(matrix(c(Data0$Y*Data0$X1,Data0$Y*Data0$X2,Data0$Y),
nrow=N,byrow=F),diag(rep(1,N)),matrix(rep(1,N),ncol=1)),cbind(odiag(rep(1,N),3),
matrix(rep(1,N+3),ncol=1)))
AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),3),matrix(rep(0,N+3),ncol=1)))
AMAT<-AMAT[-(dim(AMAT)[1]-2):-(dim(AMAT)[1]),]
bv<-matrix(c(rep(1,N),rep(0,2*N)),ncol=1)

dVEC<--C*N*matrix(c(rep(0,3),rep(1/(N*(1-alpha)),N),1),nrow=1)
DMAT<-diag(c(rep(1,2),rep(0,N+2)))
res2_EEL<-solve.QP(Dmat=as.matrix(nearPD(DMAT)$mat), dvec=dVEC,
Amat=t(AMAT), bvec=bv)
beta_hat<-res2_EEL$solution[1:3]

y_pred<-ifelse(as.matrix(Data1[,c(2,3)])%*%beta_hat[1:2]+beta_hat[3]>0,1,-1)
```

```
    CCR_EEL[k]<-mean(y_pred==Data1$Y)
  }
  ACCR_EEL[j]<-mean(CCR_EEL)
}

ACCR_EEL1<-ACCR_EEL
ACCR_EEL2<-ACCR_EEL
ACCR_EEL3<-ACCR_EEL
```

A.2.2 Transformed feature space

```
poly.feature<-function(X){
return(cbind(sqrt(2)*X[,1],sqrt(2)*X[,2],X[,1]^2,X[,2]^2,sqrt(2)*X[,1]*X[,2]))
}

# WD-based SVM
epsilon_s<-c(-3,-1)
ACCR_WD<-rep(0,length(epsilon_s))

for (j in 1:length(epsilon_s)) {
  set.seed(2024)
  CCR_WD<-rep(0,100)
  for (k in 1:100) {
    mu0<-c(0.5,-3)
    Sigma0<-diag(c(0.2,3))

    Ntotal<-5000
    N<-0.1*Ntotal  # training data size
```

```
B<-rbinom(Ntotal,1,0.5)
B0<-(B==1)-(B==0)

X0<-matrix(0,length(B0),2)
for (i in 1:length(B0)) {
  X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
}

Calpha=0.2 # proportion of contaminated data
Ncon<-Calpha*Ntotal
B_con<-rbinom(Ncon, 1, 0.5)
B0_con<-(B_con==1)-(B_con==0)
X0_con<-matrix(0,length(B0_con),2)
mu0_con<-c(2,-4)
Sigma0_con<-matrix(c(0.2,-0.5,-0.5,3),nrow=2,byrow=T)
for (i in 1:length(B0_con)) {
  X0_con[i,]<-mvrnorm(1,mu0_con,Sigma0_con)
}
B0[((1-Calpha)*Ntotal+1):Ntotal]<-B0_con
X0[((1-Calpha)*Ntotal+1):Ntotal,]<-X0_con

X0<-poly.feature(X0)

Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2],X3=X0[,3],X4=X0[,4],X5=X0[,5])

ind_t<-sample(1:Ntotal, N, replace = F)
Data0<-Data[ind_t,]  # training set
```

```
Data1<-Data[-ind_t,]  # testing set

C=100
epsilon=10^epsilon_s[j]

Data0<-as.matrix(Data0)
dVEC<-matrix(c(rep(0,6),rep(C,N),C*N*epsilon),nrow=1)
AMAT<-cbind(Data0[,1]*Data0[,c(2:6)],Data0[,1],diag(rep(1,N)),
matrix(rep(0,N),ncol=1))
AMAT<-rbind(AMAT,cbind(-Data0[,1]*Data0[,c(2:6)],-Data0[,1],
diag(rep(1,N)),matrix(rep(1,N),ncol=1)))
AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),6),matrix(rep(0,N+6),ncol=1)))
AMAT<-AMAT[-(dim(AMAT)[1]-5):-(dim(AMAT)[1]),]
bv<-matrix(c(rep(1,N),rep(1,N),rep(0,N)))

beta<-Variable(N+7)
objective<-Minimize(1/2*(norm(beta[1:5],type="2"))^2+dVEC%*%beta)
constraint1<-AMAT%*%beta>=bv
constraint2<-beta[N+7]>=norm(beta[1:5],type="2")
problem<-Problem(objective, constraints = list(constraint1, constraint2))
result_WD<-solve(problem)
beta_hat<-result_WD$getValue(beta)[1:6]
lambda_hat<-result_WD$getValue(beta)[N+7]

y_pred<-ifelse(as.matrix(Data1[,c("X1","X2","X3","X4","X5")])%*%beta_hat[1:5]
+beta_hat[6]>0,1,-1)
CCR_WD[k]<-mean(y_pred==Data1$Y)
```

```
  }
  ACCR_WD[j]<-mean(CCR_WD)
}

ACCR_WD1<-ACCR_WD
ACCR_WD2<-ACCR_WD
ACCR_WD3<-ACCR_WD

# EEL-SVM
alpha_s=c(0,0.9,0.99)
ACCR_EEL<-rep(0,length(alpha_s))

for (j in 1:length(alpha_s)) {
  set.seed(2024)
  CCR_EEL<-rep(0,100)
  for (k in 1:100) {
    mu0<-c(0.5,-3)
    Sigma0<-diag(c(0.2,3))

    Ntotal<-100
    N<-0.1*Ntotal  # training data size
    B<-rbinom(Ntotal,1,0.5)
    B0<-(B==1)-(B==0)

    X0<-matrix(0,length(B0),2)
    for (i in 1:length(B0)) {
      X0[i,]<-mvrnorm(1,mu0*B0[i],Sigma0)
    }
```

```
Calpha=0.4 # proportion of contaminated data
Ncon<-Calpha*Ntotal
B_con<-rbinom(Ncon, 1, 0.5)
B0_con<-(B_con==1)-(B_con==0)
X0_con<-matrix(0,length(B0_con),2)
mu0_con<-c(2,-4)
Sigma0_con<-matrix(c(0.2,-0.5,-0.5,3),nrow=2,byrow=T)
for (i in 1:length(B0_con)) {
  X0_con[i,]<-mvrnorm(1,mu0_con,Sigma0_con)
}
B0[((1-Calpha)*Ntotal+1):Ntotal]<-B0_con
X0[((1-Calpha)*Ntotal+1):Ntotal,]<-X0_con

X0<-poly.feature(X0)

Data<-data.frame(Y=B0, X1=X0[,1],X2=X0[,2],X3=X0[,3],X4=X0[,4],X5=X0[,5])

ind_t<-sample(1:Ntotal, N, replace = F)
Data0<-Data[ind_t,]  # training set
Data1<-Data[-ind_t,]  # testing set

C=100

alpha=alpha_s[j]
Data0<-as.matrix(Data0)
AMAT<-rbind(cbind(Data0[,1]*Data0[,c(2:6)],Data0[,1],diag(rep(1,N)),
```

```
    matrix(rep(1,N),ncol=1)),cbind(odiag(rep(1,N),6),matrix(rep(1,N+6),ncol=1)))
    AMAT<-AMAT[-(dim(AMAT)[1]-5):-(dim(AMAT)[1]),]
    AMAT<-rbind(AMAT,cbind(odiag(rep(1,N),6),matrix(rep(0,N+6),ncol=1)))
    AMAT<-AMAT[-(dim(AMAT)[1]-5):-(dim(AMAT)[1]),]
    bv<-matrix(c(rep(1,N),rep(0,2*N)),ncol=1)

    dVEC<--C*N*matrix(c(rep(0,6),rep(1/(N*(1-alpha)),N),1),nrow=1)
    DMAT<-diag(c(rep(1,5),rep(0,N+2)))
    res2_EEL<-solve.QP(Dmat=as.matrix(nearPD(DMAT)$mat), dvec=dVEC, Amat=t(AMAT),
    bvec=bv)
    beta_hat<-res2_EEL$solution[1:6]

    y_pred<-ifelse(as.matrix(Data1[,c(2:6)])%*%beta_hat[1:5]+beta_hat[6]>0,1,-1)
    CCR_EEL[k]<-mean(y_pred==Data1$Y)
  }
  ACCR_EEL[j]<-mean(CCR_EEL)
}

ACCR_EEL1<-ACCR_EEL
ACCR_EEL2<-ACCR_EEL
ACCR_EEL3<-ACCR_EEL
```