Solution Path Algorithms for Distributionally Robust Regression and Classification

by

Guangrui Tang

A Dissertation Submitted to the Faculty of the

Department of Systems and Industrial Engineering

In Partial Fulfillment of the Requirements
For the Degree of

Doctor of Philosophy

In the Graduate College

The University of Arizona

2 0 2 3

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by Guangrui Tang, titled *Solution Path Algorithms for Distributionally Robust Regression and Classification* and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

Neng Fan (Sep 15, 2023 12:10 PDT) Date: September 13, 2023

Neng Fan

Date: September 13, 2023

Jian Liu

Date: September 13, 2023

Vignesh Subbian

Date: September 13, 2023

Haiquan Li

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Neng Fan (Sep 15, 2023 12:10 PDT) Date: September 13, 2023

Neng Fan
Dissertation Committee Chair
Systems and Industrial Engineering

ACKNOWLEDGMENTS

First and foremost, I would like to express my deep appreciation to my advisor Dr. Neng Fan for his guidance, encouragement and support throughout my PhD journey. His expertise, enthusiasm and dedication to research have been instrumental in shaping my thinking and approach to the field.

I am also grateful to the members of my committee, Dr. Jian Liu, Dr. Vignesh Subbian and Dr. Haiquan Li, for their insightful comments and feedback on my work. Their expertise in the field has helped me to refine my research and improve the quality of my dissertation.

I would also like to thank my internship mentor Cong Xie at TikTok. He hosted me and gave me the opportunity to develop efficient optimizers for advertising algorithm. I learned a lot from this wonderful internship experience.

Additionally, I would like to thank the faculty and staff in the SIE department, who provided me with the resources and opportunities to pursue my academic interests. I would also like to thank everyone at Dr. Fan's group. It was great working with all you for the past four years.

Last, but not the least, I would like to thank my parents and girlfriend for their unwavering support and encouragement throughout my academic journey. Their love and understanding have been a constant source of inspiration, and I could not have completed this work without their support.

DEDICATION

To my family

Table of Contents

Table of Contents—*Continued*

Table of Contents—*Continued*

List of Figures

List of Tables

Abstract

Data uncertainty is a challenging problem in machine learning. Distributionally robust optimization (DRO) is a popular technique to address the data uncertainty problem and it minimizes the worst-case expected loss function under the ambiguity set. It has been shown that various DRO formulations can be reformulated as regularized machine learning models. The regularization hyperparameters control the bias-variance trade-off and will influence the generalization performance. However, existing works mainly focus on modeling part and solve the model by off-the-shelf solvers. To perform the hyperparameter tuning, we need to train the model multiple times and hence is computationally prohibitive for large-scale data and high-dimensional data.

Solution path algorithms provide a new approach to speed up the hyperparameter tuning. It can be proven that the optimal solutions of some machine learning models change piecewise linearly with respect to hyperparameters. Hence, we can obtain the entire solution path with respect to different hyperparameters by monitoring the breakpoints only. This piecewise linear property inspires us to develop efficient algorithms to speed up the hyperparameter tuning.

For the regression problems, we propose a general distributionally robust regression model based on DRO. The proposed model has piecewise linear loss function and elastic net penalty term. We prove the piecewise linear property of the optimal solutions to this model, which enables us to develop a solution path algorithm for the hyperparameter tuning. Doubly regularized least absolute deviations (DrLAD) regression model is proposed based this framework. A solution path algorithm is developed to speed up the tuning of two hyperparameters in this model.

For the classification problems, a new support vector machine (SVM) with double regularization terms and double margins is derived based on DRO. The proposed model can explain the data uncertainty in probabilistic way as well as perform automatic feature selection for high-dimensional data. We prove that the optimal solutions of this model change piecewise linearly with respect to the hyperparameters. Based on this piecewise linear property, a solution path algorithm is proposed to efficiently obtain the optimal solutions and thus speed up the hyperparameter tuning process. We also extend the DRO SVM to kernel space to deal with the data that are not linearly separable. A solution path algorithm is presented to efficiently solve the proposed model.

Chapter 1
Introduction

This chapter introduces the background, motivations and contributions of the dissertation. In Chapter 1.1, the motivation of the proposed research is explained after introducing the background and needs of the research. Chapter 1.2 introduces solution path algorithms and hyperparameter tuning. Chapter 1.3 states the contributions and organization of this dissertation.

1.1 Background and Motivation

The goal of supervised machine learning is to find the relationship of input $\boldsymbol{x} \in \mathbb{R}^d$ and output $y \in \mathbb{R}$ from training data. Many learning algorithms achieve this goal by solving the empirical risk minimization problem

$$\min_{\boldsymbol{\beta}} \left\{ \mathbb{E}^{\hat{\mathbb{P}}}[\ell_{\boldsymbol{\beta}}(\boldsymbol{x}, y)] = \frac{1}{n} \sum_{i=1}^{n} \ell_{\boldsymbol{\beta}}(\boldsymbol{x}_i, y_i) \right\}$$

where $\ell_{\boldsymbol{\beta}}(\boldsymbol{x}, y)$ is the loss function and $\hat{\mathbb{P}} = \frac{1}{n} \sum_{i=1}^{n} \delta_{(\boldsymbol{x}_i, y_i)}$ denotes the empirical distribution of the training data $\{(x_i, y_i)\}_{i=1}^{n}$.

However, the training data are often corrupted with noise, such as the measurement error, missing values and variability across individuals. Thus, modeling the data uncertainty is important in estimating the true input-output relationship.

Robust optimization is a classical technique to model the data uncertainty, where the data perturbations are restricted within a deterministic uncertainty set. Many machine learning models have been proposed based on robust optimization. Trafalis et al. [1] bounded the uncertainty of data by norm and proposed robust formulations of SVM for regression and classification problems. Polyhedral uncertainty sets were used by Fan et al. [2] to formulate robust SVM. Ben-Tal et al. [3] applied the Bernstein bounding scheme to model the data uncertainty. Wang et al. [4] modeled the uncertainty by chance-constraint with second-order moment information.

Different from the robust optimization, Distributionally Robust Optimization (DRO) models the data uncertainty by probabilistic metric. It minimizes the worst-case expected loss function over the ambiguity set. The ambiguity set can be constructed via moment information or probability distance. The moment ambiguity set contains all the distributions that satisfy certain moment constraints [5, 6]. But it often leads to overly conservative solutions [7]. Moment ambiguity set with low-order moments are computationally attractive but fail to converge to a singleton when the number of training samples tends to infinity [8]. Another way is to define the ambiguity sets as a ball of distributions with some probabilistic distance, such as ϕ-divergence ambiguity sets [9], Wasserstein ambiguity sets [10–12], Sinkhorn ambiguity sets [13] and so on.

In this dissertation, we mainly investigate the DRO models with Wasserstein ambiguity sets. The Wasserstein ambiguity set can contain both discrete and continuous distributions, while the Kullback–Leibler divergence distance exclude continuous distributions if the nominal distribution is discrete [10]. Fournier and Guillin [14] show that the Wasserstein ambiguity sets contain the true distribution with high confidence for a sufficiently large sample size.

Many machine learning models have been developed based on distributionally robust optimization with Wasserstein ambiguity set. Shaeezadeh-Abadeh et al. [15] proposed a distributionally robust logistic regression model. Lee and Mehrotra [16] developed a distributionally robust framework for support vector machines. Washburn et al. [17] introduced a distributionally robust formulation for ν-SVM. Chen et al. [18] developed a distributionally robust optimization approach to estimate the regression plane in a linear regression setting. Distributionally robust adversarial training in neural network was presented by Sinha et al. [19]. Shaeezadeh-Abadeh et al. [8] proposed a novel regularization technique that can be utilized in various regression and classification models based on Wasserstein DRO. Blanchet et al. [20] presented a novel robust Wasserstein profile inference based on DRO. A robust grouped variable selection model using DRO was developed by Chen and Paschalidis [21].

The Wasserstein DRO models can be reformulated as regularized machine learning models. After reformulation, the radius λ of the Wasserstein ambiguity set becomes the regularization hyperparameter. The regularization hyperparameter controls the bias-variance trade-off, and highly influences the generalization performance of the machine learning models. Thus, the hyperparameter needs to be tuned for effective learning from data.

For the distributionally robust machine learning models, most of them are solved by the off-the-self solvers. Lee and Mehrotra [16] and Luo and Mehrotra [22] proposed cutting surface based algorithms to solve distributionally robust support vector machines. Stochastic gradient descent algorithms are employed by Blanchet et al. [23] to tackle the Wasserstein DRO problems. Li et al. [24] developed an ADMM-based first-order algorithmic framework for distributionally robust logistic regression. Epigraphical projection-based incremental algorithms were proposed by Li et al. [25] to solve distributionally robust support vector machines problem.

For hyperparameter tuning, we need to solve the DRO models with different hyperparameters. The above solution techniques are proposed for solving the DRO model with fixed hyperparameters, which means that we need to train the DRO models multiple times. It is computationally intense and very inefficient. This motivates us to develop efficient algorithms to speed up the hyperparameter tuning.

1.2 Solution Path Algorithms and Hyperparameter Tuning

Hyperparameters are parameters which can define the architecture of machine learning models. The process of searching for the ideal model architecture is referred to as hyperparameter tuning.

Grid search is the commonly used method for hyperparameter tuning [26]. In grid search, we evaluate the performance for each combination of hyperparameters and select the combination which obtains the best performance. Grid search is intuitive and easy to implement. Parallel computing can be applied in grid search. It is an exhaustive search of the hyperparameter space. But it is very inefficient and the computational complexity will increase exponentially as the number of hyperparameters grows.

Random search was proposed to improve the computational efficiency of grid search [27]. In stead of directly giving a grid of values in the hyperparameter space, the hyperparameters are provided by randomly sampling from certain distributions. Random search may have better performance when the hyperparameters are not uniformly distributed. In most cases, random search is more efficient than grid search. Parallel computing can also be applied in random search. It is often used in the early stage of hyperparameter tuning to narrow down the search space. However, Random search is still a computationally intensive method.

Bayesian optimization is sequential model based optimization algorithms [28]. The information in the previous experiment can be used to improve the next experiment. A surrogate model is used to fit the observed points into the distribution of the probabilistic surrogate model. Acquisition function is used to determine the usage of different points by balancing the trade-off exploration and exploitation. In Bayesian optimization, the optimal choice of hyperparameters can be obtained by using the information in previous experiment, and running a surrogate model is cheaper than the entire objective function. So it is more efficient than grid search and random search. Due to the sequential property, Bayesian optimization is difficult to implement for parallel computing.

For some machine learning models, the optimal solutions will change piecewise linearly with respect to hyperparameters. We can obtain the entire solution path by monitoring the breakpoints only. In other words, solution path algorithms can find the optimal solutions with respect to different hyperparameters without retraining the model multiple times. Solution path algorithms are very efficient and can obtain exact solution if there is not singular problem.

For many regression models, the optimal values of the coefficients are piecewise linear with respect to hyperparameters. The Least Angle Regression (LARS) algorithm was proposed by Efron et al. [29] to fit the ordinary least squares (OLS) model and a modified version can be used to find the solution path of the Lasso model. Zou et al. [30] developed a solution path algorithm to fit elastic net regression. For machine learning models with for almost quadratic loss functions and ℓ_1-norm penalty, Rosset et al. [31] designed an efficient algorithm to fit the coefficient path. The solution path algorithm for generalized Lasso problem was developed by Tibshirani et al. [32]. λ-path [33] and ϵ-path [34] algorithms were proposed for SVR problems. Two-dimensional solution path algorithm for SVR was developed by Wang et al. [35] to tune two hyperparameters in SVR. A solution path algorithm for weighted SVR was proposed by Karasuyama et al. [36] to tune multiple hyperparameters.

For the classification problems, mainly the SVMs, the optimal values of the coefficients can also be efficiently calculated by solution path algorithms. The solution path algorithm for support vector classification (SVC) problems was developed by Hastie et al. [37], and improved algorithms [38–40] were proposed to handle the singularity problem. The solution path algorithm for the ℓ_1-norm SVM was developed by Zhu et al. [41]. The solution path algorithm for ν-SVM was designed by Gu et al. [42,43]. With the goal of tuning the two hyperparameters, the solution path algorithm for DrSVM was developed by Wang et al. [44]. The solution path algorithm for weighted SVC was proposed by Karasuyama et al. [36] to tune multiple hyperparameters.

1.3 Contributions

In this dissertation, we propose regression and classification models based on distributionally robust optimization theory. The proposed model can explain the data uncertainty and achieve good performance in high dimensional data. The hyperparameters in these models control the bias-variance trade-off. The choice of hyperparameter will influence the performance of the models. Thus, efficient solution path algorithms for these models are proposed to speed up the hyperparameter tuning.

In Chapter 2, we present a survey of solution path algorithms for regression and classification models. For each model, we present the mathematical formulation and corresponding solution path algorithms. The advantages and limitations of each model are also discussed.

In Chapter 3, we propose a general framework for regression problems based on distributionally robust optimization theory. This model can generalize many regression models, such as support vector regression, quantile regression. We also derive a new model DrLAD based on this general framework. The optimal solutions of these models are piecewise linear with respect to hyperparameters. A solution path algorithm is proposed to efficiently obtain the optimal solutions and speed up the hyperparameter tuning.

In Chapter 4, we present a new support vector machine with double regularization terms and double margins. The proposed model can explain the data uncertainty in probabilistic way as well as perform automatic feature selection for high dimensional data. We prove that the optimal solutions of this model change piecewise linearly with respect to the hyperparameters. Based on this piecewise linear property, a solution path algorithm is proposed to efficiently obtain the optimal solutions and thus speed up the hyperparameter tuning process.

In Chapter 5, we extend the distributionally robust support vector machines to kernel space. By combining the power of both DRO formulation and kernel methods, the proposed model improves the generalization performance of SVM in real-world applications. Building upon the foundation of parametric optimization, we present an efficient solution path algorithm to reduce the computational complexity and enable the application to larger data sets and more complex scenarios.

In Chapter 6, we conclude this dissertation and discuss some future research directions.

Statement of External Support and Resources In this dissertation, I used the following tools and resources: (1) Assistance from the writing skills improvement program at University of Arizona, (2) ChatGPT (GPT-3.5). A writing specialist from the writing skills improvement program reviewed and provided feedback on grammar and writing style for Chapter 2. I used ChatGPT to identify grammatical and language issues and refine sentences in the following sections of the dissertation: abstract, conclusion in Chapter 5, and conclusion in Chapter 6.

Chapter 2
Survey of Solution Path Algorithms for Machine Learning

The loss function expresses the differences between the predicted values from regression and classification models and the actual instances in machine learning. Regularization also plays an important role in machine learning, and it can mitigate overfitting problems, perform variable selection, and produce sparse models. The hyperparameter in these models controls the trade-off between the loss function and the regularization term, as well as the bias-variance trade-off. The choice of hyperparameter will influence the performance of the models. Thus, the hyperparameter needs to be tuned for effective learning from data. In some machine learning models, the optimal values for estimated coefficients are piecewise linear with respect to the hyperparameter. Efficient algorithms can be developed to compute all solutions, and these kinds of methods are called solution path algorithms. They can significantly reduce the efforts for cross-validation and highly speed up hyperparameter tuning. In this chapter, we review the solution path algorithms widely used in regression and classification machine learning problems.

The remainder of this chapter is organized as follows. Chapter 2.1 introduces the background of this chapter. Chapter 2.2 presents the solution path algorithms for regression problems. Chapter 2.3 presents the solution path algorithms for classification problems. Finally, Chapter 2.4 concludes the chapter.

This chapter is based on our published paper [45].

2.1 Introduction

Data science is an interdisciplinary field where different techniques are used to extract information and insights from data [46]. Machine learning, statistics and optimization are important techniques in data science [47, 48]. In machine learning, regression and classification models have been widely utilized in many areas, such as spam detection, face recognition, medical diagnosis, Internet of things [49, 50]. The goal of supervised learning problems is to find the relationship between input features $\mathbf{x}_i \in \mathbb{R}^m$ and response $y_i \in \mathbb{R}$ $(i = 1, \cdots, n)$ by minimizing the loss function. The regularization term is often added to balance the bias-variance trade-off and improve generalization performance. Many regularized machine learning models, such as Lasso and support vector machine (SVM), can be formulated as

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \lambda J(\boldsymbol{\beta}), \tag{2.1}$$

where $f(\mathbf{x}_i) = \boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0$, $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_m)^\intercal$ is the vector of coefficients of input features (β_0 is the intercept term, which is also considered as a coefficient in the following). Here $L(y_i, f(\mathbf{x}_i))$ is the loss function, and $J(\boldsymbol{\beta})$ is the regularization term for penalties, such as the ℓ_1-norm penalty, ℓ_2-norm penalty, elastic net penalty, and smoothly clipped absolute deviation penalty (SCAD) [51]. Additionally, $\lambda \geq 0$ is the regularization hyperparameter: $\lambda = 0$ means no regularization, while $\lambda \to +\infty$ will shrink the coefficients to zero. So hyperparameter tuning is needed to select a proper λ.

One common technique for hyperparameter tuning is cross-validation. To perform cross-validation to identify a proper value for the hyperparameter, one needs to solve the optimization model for machine learning repeatedly, a process which is inefficient and requires high computational cost. Some machine learning models, such as Lasso and SVM, can be formulated as convex linear or quadratic programming models. Thus some techniques can be applied to speed up the process of hyperparameter tuning. For example, bilevel optimization models combined with parametric programming methods are applied for hyperparameter tuning [52, 53]. Another way is to take the advantage of piecewise linear property of optimal values for coefficients in some machine learning models [31]. For convex optimization models, optimal solutions will satisfy certain optimality conditions. Based on these conditions, it can be proven that the optimal solutions will change piecewise linearly with respect to the hyperparameter. Once the change points (also called break points) can be found sequentially, all possible solutions can be obtained efficiently. These kinds of algorithms to efficiently find the optimial solutions and speed up the hyperparameter tuning are called *solution path algorithms*.

For regression problems, such as Lasso, support vector regression (SVR), and quantile regression, the response $y_i \in \mathbb{R}$ is continuous, and the loss function is $L(y_i, f(\mathbf{x}_i)) = L(y_i - f(\mathbf{x}_i))$, where $(y_i - f(\mathbf{x}_i))$ is the residual. Common loss functions for regression problems include [54]: mean squared error loss, mean absolute error loss, Huber loss, pinball loss, and ϵ-insensitive loss.

For classification problems, for example, SVM and logistic regression, $y_i \in \{+1, -1\}$ is categorical for binary classification ($y_i \in \{1, ..., k\}$ for k class classification problems), and the loss function is $L(y_i, f(\mathbf{x}_i)) = L(y_i f(\mathbf{x}_i))$, where $y_i f(\mathbf{x}_i)$ is the hyperplane. Common loss functions for classification problems include [54]: hinge loss, smooth hinge loss, exponential loss, binomial deviance loss, and cross entropy loss.

Various combinations of loss functions and penalty terms can construct different machine learning models for different application purposes. For example, the widely-used regression model Lasso considers both mean squared error loss and ℓ_1-norm penalty:

$$\min_{\beta_0, \boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0))^2 + \lambda \|\boldsymbol{\beta}\|_1.$$

The soft margin SVM model for classification considers both hinge loss and ℓ_2-norm penalty:

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n}(1 - y_i(\boldsymbol{\beta}^\mathsf{T}\mathbf{x}_i + \beta_0))_+ + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2.$$

The doubly regularized SVM (DrSVM) considers the hinge loss and elastic net penalty:

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n}(1 - y_i(\boldsymbol{\beta}^\mathsf{T}\mathbf{x}_i + \beta_0))_+\lambda_1\|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2.$$

In both Lasso and SVM models, the choice of the hyperparameter λ will strongly influence the performance of the model [29, 31, 37]. If $\lambda = 0$, there will be no regularization; if $\lambda \to +\infty$, the coefficients will be shrunk to zero. Tuning λ is very challenge due to high computation cost. Additionally, as shown in the DrSVM model, there are two tuning parameters λ_1, λ_2. For some machine learning models, the optimal values of coefficients (expressed as a function $\boldsymbol{\beta}^*(\lambda)$ of λ) are piecewise linear with respect to λ. Based on this property, some efficient algorithms can be designed to reduce the computational burden.

2.2 Solution Path Algorithms for Regression

Regression models with the input data for features $\mathbf{x}_i \in \mathbb{R}^m$ and the continuous response $y_i \in \mathbb{R}$ can be usually formulated as

$$\min_{\boldsymbol{\beta},\beta_0} \sum_{i=1}^{n} L(y_i - f(\mathbf{x}_i)) + \lambda J(\boldsymbol{\beta}),$$

where $(y_i - f(\mathbf{x}_i))$ is the residual, and λ is the regularization hyperparameter. The penalty term can take the ℓ_1-norm penalty, ℓ_2-norm penalty, elastic net penalty, and so on. The loss functions can take mean squared error loss, mean absolute error loss, Huber loss, pinball loss, ϵ-insensitive loss and so on. If the optimal solution $\boldsymbol{\beta}^*$ is piecewise linear with respect to λ, the solution path algorithm can efficiently find the entire solution path. Table 2.1 is the summary of solution path algorithms for regression models, and the names shown in this table for regression are explained in the following part of this section.

Table 2.1: Solution Path Algorithm for Regression Models

Model	Formulation	Hyperparameter	Solution path algorithm	Reference
OLS	Equation (2.2)	-	Algorithm 2.1	[29]
Lasso	Equation (2.3)	λ	Algorithm 2.2	[29]
EN regression[a]	Equation (2.6)	λ_1, λ_2	Algorithm 2.3	[30]
AQ regression[b]	Equation (2.7)	λ	Algorithm 2.4	[31]
RLAD	Equation (2.13)	λ	Algorithm 2.5	[55]
ϵ-SVR	Equation (2.22)	λ, ϵ	Algorithm 2.6 and 2.7	[33–35]
Generalized Lasso	Equation (2.33)	λ	Algorithm 2.8	[32]
WSVR	Equation (2.47)	C_i	Algorithm 2.13	[36]

[a]EN regression: Elastic net regression.
[b]AQ regression: regression model with almost quadratic loss functions and ℓ_1-norm penalty.

2.2.1 Least Angle Regression (LARS)

The Least Angle Regression (LARS) [29] is proposed to solve the OLS problems:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\intercal \boldsymbol{\beta})^2. \tag{2.2}$$

Let $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ be the input, and $\mathcal{A}$ to a subset of the indices $\{1, 2, \cdots, m\}$. Define the matrix $X_{\mathcal{A}} = [\cdots, s_j \mathbf{x}_j, \cdots]_{j \in \mathcal{A}}$, where the sign $s_j \in \{1, -1\}$. The equiangular vector is expressed as

$$\mathbf{u}_{\mathcal{A}} = X_{\mathcal{A}} \omega_{\mathcal{A}},$$

where $\omega_{\mathcal{A}} = A_{\mathcal{A}} G_{\mathcal{A}}^{-1} \mathbf{1}_{\mathcal{A}}, \mathbf{u}_{\mathcal{A}} = A_{\mathcal{A}} \mathbf{1}_{\mathcal{A}}^\intercal$ and $\|\mathbf{u}_{\mathcal{A}}\|^2 = 1$. Then $\mathcal{G}_{\mathcal{A}} = X_{\mathcal{A}}^\intercal X_{\mathcal{A}}$ and $A_{\mathcal{A}} = (\mathbf{1}_{\mathcal{A}}^\intercal \mathcal{G}_{\mathcal{A}}^\intercal \mathbf{1}_{\mathcal{A}})^{-1/2}$, where $\mathbf{1}_{\mathcal{A}}^\intercal$ is a vector of 1's of length equaling $|\mathcal{A}|$.

Let $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$, $\mathbf{y} = (y_i, \cdots, y_n)^\intercal$ and initialize $\hat{\boldsymbol{\mu}}_0 = \mathbf{0}$. Let $\hat{\boldsymbol{\mu}}_{\mathcal{A}}$ be the current LARS estimate, so the current correlation is

$$\hat{\mathbf{c}} = X^\intercal (\mathbf{y} - \hat{\boldsymbol{\mu}}_{\mathcal{A}}),$$

where the active set $\mathcal{A}$ is constrained by

$$\hat{C} = \max\{|\hat{c}_j|\} \text{ and } \mathcal{A} = \{j : |\hat{c}_j| = \hat{C}\}.$$

Let s_j be the sign of $\hat{c}_j$ and compute $X_{\mathcal{A}}, A_{\mathcal{A}}$ and $\mathbf{u}_{\mathcal{A}}$; then the inner product vector is $\mathbf{a} \equiv X'\mathbf{u}_{\mathcal{A}}$. Let $\hat{\boldsymbol{\mu}}_{\mathcal{A}_+}$ be the next predictor; then, it is updated by

$$\hat{\boldsymbol{\mu}}_{\mathcal{A}_+} = \hat{\boldsymbol{\mu}}_{\mathcal{A}} + \hat{\gamma}\mathbf{u}_{\mathcal{A}}, \quad \hat{\gamma} = \min_{j \in \mathcal{A}^c}{}^{+}\{\frac{\hat{C} - \hat{c}_j}{A_{\mathcal{A}} - a_j}, \frac{\hat{C} - \hat{c}_j}{A_{\mathcal{A}} - a_j}\},$$

where $\min^+$ means the minimum is taken over only positive components. The procedure of the LARS algorithm is shown in Algorithm 2.1.

Algorithm 2.1: Least Angle Regression

1. Normalize the data to be zero mean and unit norm. Initialize the residual $\mathbf{r} = \mathbf{y}, \beta_1, \beta_2, ..., \beta_m = 0$;
2. Calculate the correlation of $\mathbf{x}_j$ and $\mathbf{r}$ and find the most correlated $\mathbf{x}_j$;
3. Increase β_j in the direction of $sign(c_j)$ until other competitor $\mathbf{x}_k$ achieves the same correlation with the current residual;
4. Update $\mathbf{r}$ by $\mathbf{r} = \mathbf{y} - \hat{\boldsymbol{\mu}}_{\mathcal{A}}$,and increase β_j, β_k in the direction defined by the joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until $\mathbf{x}_l$ achieves the same correlation with the current residual;
5. Continue the above step until all the predictors have been entered. Stop when the correlation of $\mathbf{r}$ and $\mathbf{x}_j$ is 0. Then we obtain the full least-square solution.

2.2.2 Solution path algorithm for Lasso

The OLS model does not have good prediction accuracy due to large variance and it is not easy to interpret. Tibshirani [56] added the ℓ_1-norm penalty and obtained the Lasso model

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^{\intercal}\boldsymbol{\beta})^2 + \lambda\|\boldsymbol{\beta}\|_1, \tag{2.3}$$

where λ is the regularization hyperparameter. The ℓ_1-norm term can perform both variable selection and regularization.

The correlation in the LARS algorithm can be written as

$$\mathbf{x}_j^\intercal(\mathbf{y} - \hat{\boldsymbol{\mu}}_{\mathcal{A}}) = \hat{C} \cdot s_j, \quad \forall j \in \mathcal{A}, \tag{2.4}$$

Let $\mathcal{B}$ be the active set of variables in the solution of (2.3) for a given λ. Similar to the above equation, we have

$$\mathbf{x}_j^\intercal(\mathbf{y} - \hat{\boldsymbol{\mu}}_{\mathcal{B}}) = \hat{C} \cdot sign(\beta_j), \quad \forall j \in \mathcal{B}. \tag{2.5}$$

So in the Lasso problem, the sign of β_j should match the sign of the residual. Efron et al. [29] extended the LARS algorithm to solve the Lasso problem. The solution path algorithm for Lasso is shown in Algorithm 2.2.

Algorithm 2.2: Solution path algorithm for Lasso

1. Normalize the data to be zero mean and unit norm. Initialize the residual $\mathbf{r} = \mathbf{y}, \beta_1, \beta_2, ..., \beta_m = 0$;
2. Calculate the correlation of $\mathbf{x}_j$ and $\mathbf{r}$ and find the most correlated $\mathbf{x}_j$;
3. Increase β_j in the direction of $sign(c_j)$ until other competitor $\mathbf{x}_k$ achieves the same correlation with the current residual;
4. Update $\mathbf{r}$ by $\mathbf{r} = \mathbf{y} - \hat{\boldsymbol{\mu}}_{\mathcal{A}}$, and increase β_j, β_k in the direction defined by the joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until $\mathbf{x}_l$ achieves the same correlation with the current residual. If a non-zero coefficient hits zero, drop this variable from the active set and recompute the current joint least squares direction;
5. Continue the above step until all the predictors have been entered. Stop when the correlation of $\mathbf{r}$ and $\mathbf{x}_j$ is 0. Then we obtain the full least-squares solution.

The solution path algorithm for Lasso can obtain all the Lasso estimates using the same order of computations as a single OLS fit.

2.2.3 Solution path algorithm for Elastic net regression

Although Lasso achieves great success in many situations, it has some limitations in some scenarios. When the dimension p is larger than the number of data points n, the Lasso can select at most n features. If there is a group of variables with high pairwise correlations, the Lasso can only select one of these variables [30]. In the usual $n > p$ situation, if there are high correlations in the features, the performance of Ridge regression is better than Lasso [56]. Zou et al. [30] proposed the elastic net regression model to deal with the above scenarios

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\intercal \boldsymbol{\beta})^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{\beta}\|_2^2, \tag{2.6}$$

where λ_1, λ_2 are the hyperparameters. The elastic net penalty $\lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2$ is a combination of Lasso and ridge penalty. It can manage strongly correlated predictors and retain the good properties of Lasso and ridge penalty.

The elastic net regression can be written as a Lasso-type problem:

$$\min_{\boldsymbol{\beta}^*} \|\mathbf{y}^* - X^*\boldsymbol{\beta}^*\|_2^2 + \frac{\lambda_1}{\sqrt{1+\lambda_2}}\|\boldsymbol{\beta}\|_1,$$

where $(X^*, \mathbf{y}^*)$ are the augmented data, and $\boldsymbol{\beta}^* = \sqrt{1+\lambda_2}\boldsymbol{\beta}$. Based on the solution path algorithm for Lasso, the solution path algorithm for elastic net regression is shown in Algorithm 2.3.

Algorithm 2.3: Solution path algorithm for Elastic net regression

1. Pick a grid of values for λ_2.
2. For each λ_2, find the entire solution path with respect to λ_1 using the solution path algorithm for Lasso (see Algorithm 2.2).
3. Then the solution path for elastic net regression is obtained.

2.2.4 Solution path algorithm for regression model with almost quadratic loss functions and ℓ_1-norm penalty

Rosset et al. [31] extended the LARS algorithm [29] to a family of "almost quadratic" loss functions (e.g. squared error loss, Huber's loss) with ℓ_1-norm penalty. The optimization model is

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} L(y_i, \boldsymbol{\beta}^\intercal \mathbf{x}_i) + \lambda \|\boldsymbol{\beta}\|_1. \tag{2.7}$$

The loss function is defined to be differentiable and piecewise quadratic

$$l(y, \boldsymbol{\beta}^\intercal \mathbf{x}) = a(r)r^2 + b(r)r + c(r), \tag{2.8}$$

where $r = (y - \boldsymbol{\beta}^\intercal \mathbf{x})$ is the residual for regression and $r = (y\boldsymbol{\beta}^\intercal \mathbf{x})$ is the margin for classification. The regularized optimization problem can be rewritten as

$$\begin{aligned} \min_{\boldsymbol{\beta}^+, \boldsymbol{\beta}^-} \quad & \sum_{i=1}^{n} L(y_i, (\boldsymbol{\beta}^+ - \boldsymbol{\beta}^-)^\intercal \mathbf{x}_i) + \lambda \sum_{j=1}^{m} (\beta_j^+ + \beta_j^-) \\ s.t. \quad & \beta_j^+ \geq 0, \beta_j^- \geq 0, \quad j = 1, \cdots, m \end{aligned}$$

Let $F(\mathbf{y}, X\boldsymbol{\beta}) = \sum_i L(y_i, \boldsymbol{\beta}^\intercal \mathbf{x}_i)$, where $\mathbf{y} = (y_1, \cdots, y_n)^\intercal$. Then the corresponding KKT conditions are

$$\begin{aligned} (\nabla F(\beta))_j + \lambda - \lambda_j^+ = 0, \quad \lambda_j^+ \beta_j^+ = 0, \\ -(\nabla F(\beta))_j + \lambda - \lambda_j^- = 0, \quad \lambda_j^- \beta_j^- = 0, \end{aligned}$$

where $\lambda_j^+ \geq 0$ and $\lambda_j^- \geq 0$ are Lagrange multipliers. Based on KKT conditions, we can find that variables can have nonzero coefficients when $|\nabla F(\hat{\beta}(\lambda))_j| = \lambda$. Then, when λ is fixed, for the active set $\mathcal{A} = \{j : \hat{\beta}_j(\lambda) \neq 0\}$, we have

$$j \in \mathcal{A} \Rightarrow |\nabla F(\hat{\beta}(\lambda))_j| = \lambda, \quad sign(\nabla F(\hat{\beta}(\lambda))_j) = -sign(\hat{\beta}(\lambda)_j), \tag{2.11a}$$

$$j \notin \mathcal{A} \Rightarrow |\nabla F(\hat{\beta}(\lambda))_j| \leq \lambda. \tag{2.11b}$$

When λ changes, the direction in which $\hat{\beta}(\lambda)$ is moving is determined by (2.11). So when the loss is twice differentiable and penalty is right differentiable, we can obtain

$$\frac{\partial \hat{\beta}(\lambda)_{\mathcal{A}}}{\partial \lambda} = -(\nabla^2 F(\hat{\beta}(\lambda))_{\mathcal{A}})^{-1} \cdot sign(\hat{\beta}(\lambda)_{\mathcal{A}}), \tag{2.12}$$

Then for the almost quadratic loss functions, we have

$$\nabla^2 F(\hat{\beta}(\lambda))_{\mathcal{A}} = \sum_i 2a(r(y_i, \hat{\beta}(\lambda)_{\mathcal{A}}^\intercal \mathbf{x}_{\mathcal{A}i}))\mathbf{x}_{\mathcal{A}i}\mathbf{x}_{\mathcal{A}i}^\intercal.$$

As $a(\cdot)$ is a piecewise constant function, the solution path will be piecewise linear. The direction in (2.12) will change when one of these events happens:

- Add a variable: when $\hat{\beta}(\lambda)$ keeps moving, $|\nabla F(\hat{\beta}(\lambda))_{\mathcal{A}^C}| \leq \lambda$ will not hold, so a new variable should join $\mathcal{A}$.
- Drop a variable: when we continue the same direction, $sign(\nabla F(\hat{\beta}(\lambda))_j) = -sign(\hat{\beta}(\lambda)_j)$ will not hold because a coefficient in $\mathcal{A}$ hits 0. So we need to drop this coefficient.
- Cross a knot: A "generalized residual" $r(y_i, \hat{\beta}(\lambda)^\intercal \mathbf{x}_i)$ hits a non-twice differentiability point in F.

When the above events happen, $\mathcal{A}$ needs to be updated and the direction in (2.12) needs to be recalculated. The algorithm is summarized in Algorithm 2.4.

Algorithm 2.4: Solution path algorithm for regression model with almost quadratic loss functions and ℓ_1-norm penalty

1 Initialization: $\beta = 0, \mathcal{A} = \arg\max_j |\nabla F(\beta)|_j, \gamma_{\mathcal{A}} = -sign(\nabla F(\beta))_{\mathcal{A}}, \gamma_{\mathcal{A}^C} = 0$;

2 Calculate the step length:
$d_1 = \min\{d > 0 : |\nabla F(\beta + d\gamma)_j| = |\nabla F(w\beta + d\gamma)_{\mathcal{A}}|, j \notin \mathcal{A}\}$,
$d_2 = \min\{d > 0 : (\beta + d\gamma)_j = 0, j \in \mathcal{A}\}$(hit 0),
$d_3 = \min\{d > 0 : r(y_i, (\beta + d\gamma)^\intercal x_i)$ hits a "knot", $i = 1, \cdots, n\}$, Find step length: $d = \min(d_1, d_2, d_3)$, and update the step by $\beta \leftarrow \beta + d\gamma$;

3 If $d = d_1$ then add variable attaining equality at d to $\mathcal{A}$, If $d = d_2$ then remove variable attaining 0 at d to $\mathcal{A}$, If $d = d_3$ for i^* then assign new $a(r(y_{i^*}, \beta^\intercal \mathbf{x}_{i^*}))$ from (2.8);

4 Calculate new direction: $C = \sum_i a(r(y_{i^*}, \beta^\intercal \mathbf{x}_i))\mathbf{x}_{\mathcal{A},i}\mathbf{x}_{\mathcal{A},i}^\intercal$,
$\gamma_{\mathcal{A}} = C^{-1} \cdot sign(\beta_{\mathcal{A}})$ and $\gamma_{\mathcal{A}^C} = 0$.

2.2.5 Solution path algorithm for regularized least absolute deviations regression

Wang et al. [55] proposed a solution path algorithm for the regularized least absolute deviations (LAD) regression model. The LAD loss is robust to noise, and the ℓ_1-norm penalty provides an automatic feature selection function. The model is formulated as

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n} |y_i - (\beta_0 + \mathbf{x}_i^{\intercal}\boldsymbol{\beta})| + \lambda\|\boldsymbol{\beta}\|_1, \tag{2.13}$$

where λ is the regularization hyperparameter, it controls the compromise between the LAD loss and ℓ_1-norm penalty. This problem is equivalent to

$$\min_{\beta_0,\boldsymbol{\beta},\boldsymbol{\epsilon}} \sum_{i=1}^{n} \epsilon_i \tag{2.14a}$$

$$s.t.\ y_i - (\beta_0 + \mathbf{x}_i^{\intercal}\boldsymbol{\beta}) \leq \epsilon_i, \quad i = 1, \cdots, n \tag{2.14b}$$

$$(\beta_0 + \mathbf{x}_i^{\intercal}\boldsymbol{\beta}) - y_i \leq \epsilon_i, \quad i = 1, \cdots, n \tag{2.14c}$$

$$\sum_{i=1}^{n} sign(\beta_i)\beta_i \leq s \tag{2.14d}$$

$$\epsilon_i \geq 0, \quad i = 1, \cdots, n \tag{2.14e}$$

where $s > 0$ is the regularization hyperparameter, it controls the size of the coefficient. For every s in (2.14), we can find a corresponding λ in (2.14). The KKT conditions of (2.14) are

$$\begin{aligned}
-\sum_{i=1}^{n}(\eta_i^+ - \eta_i^-) &= 0 \\
-\sum_{i=1}^{n}(\eta_i^+ - \eta_i^-)x_{ij} + \lambda \cdot sign(\beta_j) &= 0, \quad \forall \beta_j \neq 0 \\
1 - \eta_i^+ - \eta_i^- - \gamma_i &= 0, \quad i = 1, \cdots, n \\
\eta_i^+(y_i - (\beta_0 + \mathbf{x}_i^{\intercal}\boldsymbol{\beta}) - \epsilon_i) &= 0, \quad i = 1, \cdots, n \\
\eta_i^-(y_i - (\beta_0 + \mathbf{x}_i^{\intercal}\boldsymbol{\beta}) + \epsilon_i) &= 0, \quad i = 1, \cdots, n \\
\lambda(\sum_{i=1}^{n} \beta_i \cdot sign(\beta_i) - s) &= 0,
\end{aligned}$$

$$\gamma_i \epsilon_i = 0, \quad i = 1, \cdots, n$$

where $\eta_i^+, \eta_i^-, \gamma_i, \lambda$ are Lagrange multipliers. Based on the KKT conditions, we can obtain the following sets:

- $\mathcal{R} = \{i : y_i - (\beta_0 + \mathbf{x}_i^\intercal \boldsymbol{\beta}) > 0, \eta_i^+ = 1, \eta_i^- = 0\}$
- $\mathcal{E} = \{i : y_i - (\beta_0 + \mathbf{x}_i^\intercal \boldsymbol{\beta}) = 0, \eta_i^+ + \eta_i^- \leq 1\}$
- $\mathcal{L} = \{i : y_i - (\beta_0 + \mathbf{x}_i^\intercal \boldsymbol{\beta}) < 0, \eta_i^+ = 0, \eta_i^- = 1\}$
- $\mathcal{V} = \{j : \beta_j \neq 0\}$

For points in $\mathcal{E}$ and $\mathcal{V}$, we can obtain the following system

$$y_i - (\beta_0 + \sum_{j \in \mathcal{V}} x_{ij} \beta_j) = 0, \quad i \in \mathcal{E},$$
$$\sum_{j \in \mathcal{V}} \beta_j \cdot sign(\beta_j) - s = 0.$$

Take derivatives with respect to s, then

$$\frac{d\beta_0}{ds} + \sum_{j \in \mathcal{V}} x_{ij} \frac{d\beta_j}{ds} = 0, \quad i \in \mathcal{E}$$
$$\sum_{j \in \mathcal{V}} sign(\beta_i) \frac{d\beta_j}{ds} = 1.$$

As s increases, the residuals of points in $\mathcal{R}$ or $\mathcal{L}$ might reach 0. It will happen at

$$\Delta s^1 = \min(\{\Delta s_i^1 | i \in \mathcal{L} \cup \mathcal{R}\}), \tag{2.18}$$

where $\Delta s_i^1 = (y_i - (\beta_0 + \sum_{j \in \mathcal{V}} x_{ij} \beta_j))/(\frac{d\beta_0}{ds} + \sum_{j \in \mathcal{V}} x_{ij} \frac{d\beta_j}{ds})$. Another event is that an active variable in $\mathcal{V}$ becomes inactive. It will happen at

$$\Delta s^2 = \min(\{\Delta s_j^2 | j \in \mathcal{V}\}),$$

where $\Delta s_j^2 = -\beta_j/(\frac{d\beta_j}{ds})$. Then the overall step size is

$$\Delta s = \min(\{\Delta s^1, \Delta s^2\}). \tag{2.19}$$

Let $\mathcal{E}^{\text{new}}, \mathcal{V}^{\text{new}}, \mathcal{R}^{\text{new}}, \mathcal{L}^{\text{new}}$ be the new set after the above events happened. In order to keep $|\mathcal{E}^{\text{new}}| = |\mathcal{V}^{\text{new}}|$, we need to either remove a point from $\mathcal{E}^{\text{new}}$ or add a new variable to $\mathcal{E}^{\text{new}}$. It happens at

$$\Delta\lambda_i = \min(\{\Delta\lambda_i^1, \Delta\lambda_i^2\}),$$

where $\Delta\lambda_i^1 = (1-\eta_i^{\text{old}})/\frac{d\eta_i}{d\lambda}, \Delta\lambda_i^2 = (-1-\eta_i^{\text{old}})/\frac{d\eta_i}{d\lambda}$. When the inactive variable become active, we can obtain

$$\Delta\tilde{\lambda}_j^1 = \frac{\sum_{i\in\mathcal{E}^{\text{new}}} \eta_i^{\text{old}} x_{ij} + \sum_{i\in\mathcal{R}^{\text{new}}} x_{ij} - \sum_{i\in\mathcal{L}^{\text{new}}} x_{ij} - \lambda}{1 - \sum_{i\in\mathcal{E}^{\text{new}}} \frac{d\eta_i}{d\lambda} x_{ij}},$$
$$\Delta\tilde{\lambda}_j^2 = \frac{\sum_{i\in\mathcal{E}^{\text{new}}} \eta_i^{\text{old}} x_{ij} + \sum_{i\in\mathcal{R}^{\text{new}}} x_{ij} - \sum_{i\in\mathcal{L}^{\text{new}}} x_{ij} + \lambda}{-1 - \sum_{i\in\mathcal{E}^{\text{new}}} \frac{d\eta_i}{d\lambda} x_{ij}}.$$

The sign of β_j is calculated by

$$sign(\beta_j) = \begin{cases} +1, & \Delta\tilde{\lambda}_j = \Delta\tilde{\lambda}_j^1 \\ -1, & \Delta\tilde{\lambda}_j = \Delta\tilde{\lambda}_j^2 \end{cases}, \Delta\tilde{\lambda}_j = \begin{cases} \Delta\tilde{\lambda}_j^1, & \Delta\tilde{\lambda}_j^2 > 0 \\ \Delta\tilde{\lambda}_j^2, & \Delta\tilde{\lambda}_j^1 > 0 \\ \max\{\Delta\tilde{\lambda}_j^1, \Delta\tilde{\lambda}_j^1\}, & \text{otherwise} \end{cases}$$

Then the step size of $\Delta\lambda$ is

$$\Delta\lambda = \max(\{\Delta\lambda_i | i \in \mathcal{E}^{\text{new}}\}, \{\Delta\tilde{\lambda}_j | j \notin \mathcal{V}\}). \tag{2.21}$$

The solution path algorithm for the regularized LAD regression is shown in Algorithm 2.5.

2.2.6 Solution path algorithm for ϵ-SVR

λ-path algorithm for ϵ-SVR

Previous methods solve the SVR for a prefixed regularization hyperparameter, such as interior point method [57, 58], subset selection method [59], sequential minimal

Algorithm 2.5: Solution path algorithm for regularized LAD regression

1 Initialization: $\lambda, s, , \eta_i, \beta_0, \beta_j, \mathcal{L}, \mathcal{R}, \mathcal{E}$;
2 Calculate Δs using (2.19), update $s, \beta_0, \beta_j, \mathcal{L}, \mathcal{R}, \mathcal{E}$;
3 Calculate $\Delta\lambda$ using (2.21), update $\lambda, \beta_0, \beta_j, \mathcal{L}, \mathcal{R}, \mathcal{E}$;
4 Stop the algorithm if one of conditions is met: (i) $\lambda = 0$, (ii) Reach the maximal iteration, and (iii) All the predictor variables are in $\mathcal{V}$.

optimization method and so on. Gunter et al. [33] proposed an algorithm which can compute the entire solution path of the ϵ-SVR with respect to λ. The ϵ-SVR [58, 60] is formulated as

$$\min_{\beta_0, \boldsymbol{\beta}, \xi^{(*)}} \quad \frac{\lambda}{2}\|\boldsymbol{\beta}\|^2 + \sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{2.22a}$$

$$s.t. \quad y_i - (\langle \boldsymbol{\beta}, \phi(\mathbf{x}_i)\rangle + \beta_0) \leq \epsilon + \xi_i \tag{2.22b}$$

$$(\langle \boldsymbol{\beta}, \phi(\mathbf{x}_i)\rangle + \beta_0) - y_i \leq \epsilon + \xi_i^* \tag{2.22c}$$

$$\xi_i^{(*)} \geq 0, \quad i = 1, ..., n \tag{2.22d}$$

where $(*)$ denote both the variables with and without asterisks. $\phi(\cdot)$ is the nonlinear feature map to some kernel space. λ is the regularization hyperparameter, controlling the trade-off between ϵ-insensitive loss and the complexity of the fitted model. ϵ is the hyperparameter in ϵ-insensitive loss. It defines two margins around the hyperplane, where no penalty is given to points between two margins. The dual problem is formulated as

$$\min_{\boldsymbol{\alpha}^*} \quad \frac{1}{2\lambda}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\mathbf{x}_i, \mathbf{x}_j)$$

$$+ \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)y_i \tag{2.23a}$$

$$s.t. \quad 0 \leq \alpha_i^{(*)} \leq 1 \tag{2.23b}$$

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0, \quad i = 1, ..., n \tag{2.23c}$$

Then the regression function can be written as

$$f(\mathbf{x}) = \langle \boldsymbol{\beta}, \phi(\mathbf{x}) \rangle + \beta_0 = \frac{1}{\lambda} \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}) + \beta_0,$$

Based on KKT conditions, data points are partitioned into the following sets:

- $\mathcal{R} = \{i : y_i - f(x_i) > \epsilon, \alpha_i = 1, \alpha_i^* = 0\}$(right of the elbows)
- $\mathcal{E_R} = \{i : y_i - f(x_i) = \epsilon, 0 \leq \alpha_i = 1 \leq 1, \alpha_i^* = 0\}$(right elbow)
- $\mathcal{C} = \{i : -\epsilon < y_i - f(x_i) < \epsilon, \alpha_i = 0, \alpha_i^* = 0\}$(center)
- $\mathcal{E_L} = \{i : y_i - f(x_i) = -\epsilon, \alpha_i = 0, 0 \leq \alpha_i^* \leq 1\}$(left elbow)
- $\mathcal{L} = \{i : y_i - f(x_i) < -\epsilon, \alpha_i = 0, \alpha_i^* = 1\}$(left of the elbows)

Since β_0 is not unique, it is initialized to be one of the boundary values to keep one point at an elbow. The above sets will remain the same until one of the following events happens:

- A point from $\mathcal{C}$ moves into $\mathcal{E_R}$ or $\mathcal{E_L}$, with $\alpha_i = 0$ or $\alpha_i^* = 0$.
- A point from $\mathcal{R}$ or $\mathcal{L}$ moves into $\mathcal{E_R}$ or $\mathcal{E_L}$, with $\alpha_i = 1$ or $\alpha_i^* = 1$.
- A point from $\mathcal{E_R}$ or $\mathcal{E_L}$ moves into $\mathcal{C}$, with $\alpha_i = 0$ or $\alpha_i^* = 0$.
- A point from $\mathcal{E_R}$ or $\mathcal{E_L}$ moves into $\mathcal{R}$ or $\mathcal{L}$, with $\alpha_i = 1$ or $\alpha_i^* = 1$.

The parameters after the ℓth event are denoted as θ_i^ℓ, β_0^ℓ and λ^ℓ. Let $\alpha_0 = \lambda \cdot \beta_0$, then for $\lambda^{\ell+1} < \lambda < \lambda^\ell$, we have

$$f(x) = \frac{1}{\lambda}[(\alpha_0 - \alpha_0^\ell) + \sum_{i \in \mathcal{E}_\mathcal{R}^\ell} (\alpha_i - \alpha_i^\ell) K(x_i, x) - \sum_{j \in \mathcal{E}_\mathcal{L}^\ell} (\alpha_j^* - \alpha^{*\ell}_j) K(x_j, x) + \lambda^\ell f^\ell(x)].$$

Let $v_i = \alpha_i - \alpha_i^\ell$, $v_j = \alpha_j^* - \alpha_j^{*\ell}$, $v_0 = \alpha_0 - \alpha_0^\ell$, for the points in $\mathcal{E}_\mathcal{R}^\ell$ and $\mathcal{E}_\mathcal{L}^\ell$, we have

$$\sum_{i \in \mathcal{E}_\mathcal{R}^\ell} v_i K(x_i, x_k) - \sum_{j \in \mathcal{E}_\mathcal{L}^\ell} v_j^* K(x_j, x_k) + v_0 = (\lambda - \lambda^\ell)(y_k - \epsilon), \forall k \in \mathcal{E}_\mathcal{R}^\ell \tag{2.26a}$$

$$\sum_{i \in \mathcal{E}_\mathcal{R}^\ell} v_i K(x_i, x_m) - \sum_{j \in \mathcal{E}_\mathcal{L}^\ell} v_j^* K(x_j, x_m) + v_0 = (\lambda - \lambda^\ell)(y_m + \epsilon), \forall m \in \mathcal{E}_\mathcal{L}^\ell \tag{2.26b}$$

Based on the dual problem (2.23), we can obtain

$$\sum_{i \in \mathcal{E}_\mathcal{R}^\ell} v_i - \sum_{j \in \mathcal{E}_\mathcal{R}^\ell} v_j^* = 0, \tag{2.27}$$

Let $\mathbf{K}_\ell$ be the kernel sub-matrix for the points at the elbows, $\mathbf{v} = (v_i)\ \forall i \in \mathcal{E}_\mathcal{R}^\ell$, $\mathbf{v}^* = (v_j^*)\ \forall j \in \mathcal{E}_\mathcal{L}^\ell$, $\mathbf{y}_R^\ell = (y_i - \epsilon)\ \forall i \in \mathcal{E}_\mathcal{R}^\ell$, $\mathbf{y}_L^\ell = (y_j + \epsilon)\ \forall i \in \mathcal{E}_\mathcal{L}^\ell$, then we obtain the following matrix:

$$\mathbf{A}_\ell = \begin{bmatrix} 0 & \mathbf{1}^\intercal \\ \mathbf{1} & \mathbf{K}_\ell \end{bmatrix}, \quad \mathbf{v}^a = \begin{bmatrix} v_0 \\ \mathbf{v} \\ -\mathbf{v}^* \end{bmatrix}, \quad \mathbf{y}^a = \begin{bmatrix} 0 \\ \mathbf{y}_R^\ell \\ \mathbf{y}_L^\ell \end{bmatrix},$$

Then, we can obtain $\mathbf{A}_\ell \mathbf{v}^a = (\lambda - \lambda^\ell)\mathbf{y}^a$. If $\mathbf{A}_\ell$ is of full rank, let $\mathbf{b} = \mathbf{A}_\ell^{-1}\mathbf{y}^a$, we will achieve the following linear relationship:

$$\alpha_0 = \alpha_0^\ell + (\lambda - \lambda^\ell) b_0, \tag{2.28a}$$

$$\alpha_i = \alpha_i^\ell + (\lambda - \lambda^\ell) b_i, \quad \forall i \in \mathcal{E}_\mathcal{R}^\ell \tag{2.28b}$$

$$\alpha_j^* = \alpha_j^{*\ell} - (\lambda - \lambda^\ell) b_j, \quad \forall j \in \mathcal{E}_\mathcal{L}^\ell \tag{2.28c}$$

Hence, for $\lambda^{\ell+1} < \lambda < \lambda_\ell$, the $\alpha_i^{(*)}$ and α_0 for points at the $\mathcal{E}_\mathcal{R}^\ell, \mathcal{E}_\mathcal{L}^\ell$ proceed linearly in λ. And we have

$$f(x) = \frac{\lambda^\ell}{\lambda}[f^\ell(x) - g^\ell(x)] + g^\ell(x), \tag{2.29}$$

where $g^\ell(x) = b_0 + \sum_{i \in \mathcal{E}_\mathcal{R}^\ell} b_i K(x_i, x) - \sum_{j \in \mathcal{E}_\mathcal{L}^\ell} b_j K(x_j, x)$.

When the point in the elbow moves to the boundary, $\lambda^{\ell+1}$ can be computed by equations (2.28) and (2.29) for given λ^ℓ.

When the point moves from $\mathcal{L}$, $\mathcal{R}$ or $\mathcal{C}$ to the elbow, λ is calculated by

$$\lambda = \lambda^{\ell}(\frac{f^{\ell}(x_j) - g^{\ell}(x_j)}{y_j - g^{\ell}(x_j) - \epsilon}), \forall j \in (\mathcal{R}^{\ell} \cup \mathcal{C}^{\ell}), \tag{2.30a}$$

$$\lambda = \lambda^{\ell}(\frac{f^{\ell}(x_j) - g^{\ell}(x_j)}{y_j - g^{\ell}(x_j) + \epsilon}), \forall j \in (\mathcal{L}^{\ell} \cup \mathcal{C}^{\ell}), \tag{2.30b}$$

where $\lambda^{\ell+1}$ is the largest $\lambda < \lambda^{\ell}$.

When λ becomes sufficiently small to zero or when the sets $\mathcal{R}$ and $\mathcal{L}$ become empty, the algorithm will terminate. In the former case, $f^{\ell} - g^{\ell}$ should be sufficiently small as well. The λ-path algorithm is shown in Algorithm 2.6.

Algorithm 2.6: λ-path algorithm for ϵ-SVR

1 Initialize λ, α, α^* α_0, $\mathcal{L}$, $\mathcal{E}_{\mathcal{L}}$, $\mathcal{C}$, $\mathcal{E}_{\mathcal{R}}$, $\mathcal{R}$;
2 Find the break point λ_{ℓ} where an event happens using (2.28), (2.30);
3 Update $\mathcal{L}$, $\mathcal{E}_{\mathcal{L}}$, $\mathcal{C}$, $\mathcal{E}_{\mathcal{R}}$, $\mathcal{R}$, and update α, α^* α_0 and $f(x)$ by equation (2.28) and (2.29);
4 Find the break points using (2.28) and (2.30), and choose the largest $\lambda < \lambda^{\ell}$ to be $\lambda^{\ell+1}$. Then return to step 3;
5 End the algorithm if $\mathcal{L}$ and $\mathcal{R}$ are empty or λ is sufficiently small.

This algorithm can compute the solution path for some fixed ϵ values. But if there is only one or even no point at the boundaries, the tube has to move and rotate until two valid points enter the tube. Because there are many possible combinations for the two points, it is hard to realize this algorithm.

ϵ-path algorithm for ϵ-SVR.

To overcome the limitations in the solution path algorithm for ϵ-SVR, Wang et al. [34, 35] proposed the ϵ-path algorithm.

Similar to Section 2.2.6, the data points are partitioned into $\mathcal{L}, \mathcal{E}_{\mathcal{L}}, \mathcal{C}, \mathcal{E}_{\mathcal{R}}, \mathcal{R}$, and the break points correspond to the same events in Section 2.2.6. For the points inside or outside the tube, the $\alpha_i^{(*)}$ will remain fixed until an event happens.

For initialization, if $\epsilon = \infty$, we can easily solve problem (2.23), and the solution is $\alpha_i^{(*)} = 0$. It is much easier than the λ-path algorithm in Section 2.2.6, where we need to solve a linear programming problem.

Assume λ is fixed, and let ϵ^ℓ denote the value of ϵ after ℓth event happened. Then ϵ is reduced until each elbow contains at least one point, which means $|\mathcal{E}_\mathcal{R}| > 0$ and $|\mathcal{E}_\mathcal{L}| > 0$. When $\epsilon^{\ell+1} < \epsilon < \epsilon^\ell$, the regression function can be written as

$$f(\mathbf{x}) = \frac{1}{\lambda}\Big[\sum_{i\in\mathcal{E}_\mathcal{R}^\ell}(\alpha_i - \alpha_i^\ell)K(\mathbf{x}_i,\mathbf{x}) - \sum_{j\in\mathcal{E}_\mathcal{L}^\ell}(\alpha_i^* - \alpha_i^{*\ell})K(\mathbf{x}_j,\mathbf{x})\Big] + f^\ell(x).$$

Following the same procedure in Section 2.2.6, we can obtain $\mathbf{A}^\ell\mathbf{v}_0 = \lambda(\epsilon^\ell - \epsilon)\mathbf{1}^a$. If $\mathbf{A}_l$ has full rank, let $\mathbf{b} = (\mathbf{A}_\ell)^{-1}\mathbf{1}^a$, we will have the following linear relationship:

$$\alpha_0 = \alpha_0^\ell + \lambda(\epsilon^\ell - \epsilon)b_0, \tag{2.31a}$$

$$\alpha_i = \alpha_i^\ell + \lambda(\epsilon^\ell - \epsilon)b_i, \quad \forall i \in \mathcal{E}_\mathcal{R}^\ell \tag{2.31b}$$

$$\alpha_j^* = \alpha_j^{*\ell} - \lambda(\epsilon^\ell - \epsilon)b_j, \quad \forall j \in \mathcal{E}_\mathcal{L}^\ell \tag{2.31c}$$

Hence, for $\epsilon^{\ell+1} < \epsilon < \epsilon^\ell$, the $\alpha_i^{(*)}$ and α_0 for points at the elbow proceed linearly in ϵ. And we have

$$f(x) = (\epsilon^\ell - \epsilon)h^\ell(\mathbf{x}) + f^\ell(\mathbf{x}), \tag{2.32}$$

where $h^\ell(\mathbf{x}) = \sum_{i\in\mathcal{E}_\mathcal{R}^\ell} b_iK(\mathbf{x}_i,\mathbf{x}) - \sum_{j\in\mathcal{E}_\mathcal{L}^\ell} b_jK(\mathbf{x}_j,\mathbf{x}) + b_0$.

As ϵ decreases, the Lagrange multipliers change until an event happens. Then, we find the largest $\epsilon < \epsilon^\ell$, and then assign it as $\epsilon^{\ell+1}$ and update the point sets. The ϵ-path algorithm is in Algorithm 2.7.

The λ-path algorithm and ϵ-path algorithm can be combined together to explore the two dimensional solution space [35]. In the ϵ-path algorithm, when there is at least one point in each elbow, the λ-path update can be performed. A complicated situation arises when some $\alpha_i^{(*)}$ in elbow point is equal to 0 or 1. Because λ can either increase or decrease, whether this point should be included into the updating set is

Algorithm 2.7: ϵ-path algorithm for ϵ-SVR

1 Initialize ϵ, α, α^* α_0, $\mathcal{L}$, $\mathcal{E}_{\mathcal{L}}$, $\mathcal{C}$, $\mathcal{E}_{\mathcal{R}}$, $\mathcal{R}$;
2 Find the break point λ_ℓ where an event happens using (2.31), (2.32);
3 Update $\mathcal{L}$, $\mathcal{E}_{\mathcal{L}}$, $\mathcal{C}$, $\mathcal{E}_{\mathcal{R}}$, $\mathcal{R}$, and update α, α^* α_0 and $f(x)$ by equation (2.31) and (2.32);
4 Find the break points using (2.31) and (2.32), and choose the largest $\epsilon < \epsilon^\ell$ to be $\epsilon^{\ell+1}$. Then return to step 3;
5 End the algorithm if $\mathcal{L}$ and $\mathcal{R}$ are empty or ϵ is sufficiently small.

uncertain. One way to deal with it is to terminate the ϵ-path algorithm between two consecutive events. Then $\alpha_i^{(*)}$ in elbow point will not be 0 or 1, and so the λ-path update can proceed successfully.

2.2.7 Solution path algorithm for generalized Lasso

Based on the LARS algorithm, Tibshirani and Taylor [32] proposed a path algorithm for the generalized Lasso problem. This problem has the ℓ_1-norm penalty of the product of a matrix D and coefficient vector. The generalized Lasso problems can be formulated as

$$\min_{\boldsymbol{\beta}} \; \frac{1}{2}\|\mathbf{y} - X\boldsymbol{\beta}\|_2^2 + \lambda\|D\boldsymbol{\beta}\|_1, \tag{2.33}$$

where $X \in \mathbb{R}^{n\times m}$ is a general design matrix and $D \in \mathbb{R}^{m\times p}$ is a specified penalty matrix. Different choices of D can lead to different models: the fused lasso, trend filtering, wavelet smoothing and a method for outlier detection. Here λ is the regularization hyperparameter, and it controls the trade-off between loss function and the complexity of the fitted model.

Let $X = I$, $\mathbf{z} = D\boldsymbol{\beta}$, the problem can be written as

$$\begin{aligned} \min_{\boldsymbol{\beta},\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{y} - \boldsymbol{\beta}\|_2^2 + \lambda\|\mathbf{z}\|_1 \\ s.t. \quad & D\boldsymbol{\beta} = \mathbf{z}. \end{aligned}$$

The dual problem is

$$\min_{u} \quad \frac{1}{2}\|\mathbf{y} - D^{\intercal}\mathbf{u}\|_2^2 \tag{2.35a}$$

$$s.t. \quad \|\mathbf{u}\|_{\infty} \leq \lambda, \tag{2.35b}$$

where the constraint is simply a box, free of any linear transformation.

For $\lambda \leq \lambda_k$, the "hitting time" is when the path hits the boundary. Similarly, the "leaving time" is when the path leaves the boundary. The coordinate with the largest hitting time is called the "hitting coordinate", and the coordinate with the largest leaving time is called the "leaving coordinate". $\mathcal{B}$ is the list of boundary coordinates, and s contains the corresponding signs. Then, we can define for a matrix A and a vector x:

$$A_{\mathcal{B}} = \begin{bmatrix} A_{i_1} \\ \vdots \\ A_{i_k} \end{bmatrix}, \quad x_{\mathcal{B}} = (x_{i_1}, \cdots, x_{i_k})^{\intercal}$$

where $A_{\mathcal{B}}$ indexes the rows of A that are in $\mathcal{B}$. The subscript $-\mathcal{B}$ is used to index elements which are not in $\mathcal{B}$, such as $A_{-\mathcal{B}}, x_{-\mathcal{B}}$.

For the above dual problem (2.35), the KKT conditions are

$$(DD^{\intercal}\mathbf{u})_i - (D\mathbf{y})_i + \alpha\boldsymbol{\gamma}_i = 0 \quad i = 1, ..., m,$$

where $\mathbf{u}, \alpha, \boldsymbol{\gamma}$ are subject to the following constraints:

$$\|\mathbf{u}\|_{\infty} \leq \lambda, \tag{2.37a}$$

$$\alpha \geq 0, \tag{2.37b}$$

$$\alpha \cdot (\|\mathbf{u}\|_{\infty} - \lambda) = 0, \tag{2.37c}$$

$$\|\boldsymbol{\gamma}\|_1 \leq 1, \tag{2.37d}$$

$$\boldsymbol{\gamma}^{\intercal}\mathbf{u} = \|\mathbf{u}\|_{\infty}. \tag{2.37e}$$

The hitting times correspond to events in which equation (2.37a) is violated, and leaving times correspond to events in which equations (2.37a)-(2.37e) are violated.

At the kth iteration, the boundary set is $\mathcal{B} = \mathcal{B}(\lambda_k)$, and corresponding signs are $s = s(\lambda_k)$. By the boundary lemma [32], the solution satisfies $\hat{u}_{\lambda_k,\mathcal{B}} = \lambda_k s$, for any $\lambda \in [0, \lambda_k]$.

For $\lambda = \lambda_k$, the solution is $\hat{u}_{\lambda_k,\mathcal{B}} = \lambda_k s$. For $\lambda < \lambda_k$, problem (2.35) can be reduced to

$$\begin{aligned} \min_{u_{-\mathcal{B}}} \ & \frac{1}{2}\|\mathbf{y} - \lambda(D_{\mathcal{B}})^{\intercal} - (D_{-\mathcal{B}})^{\intercal}\mathbf{u}_{-\mathcal{B}}\|_2^2 \\ s.t. \ & \|\mathbf{u}_{-\mathcal{B}}\|_\infty \leq \lambda \end{aligned}$$

The solution of the above problem is the least squares estimate

$$\hat{\mathbf{u}}_{\lambda_k,-\mathcal{B}} = (D_{-\mathcal{B}}(D_{-\mathcal{B}})^T)^+ D_{-\mathcal{B}}(y - \lambda_k(D_{-\mathcal{B}})^T s), \tag{2.39}$$

where A^+ denotes the (Moore-Penrose) pseudoinverse of a matrix A. To simplify the notation, the above solution can be denoted as $\hat{\mathbf{u}}_{\lambda_k,-\mathcal{B}} = a - \lambda_k b$. The solution will continue to be $\hat{\mathbf{u}}_{\lambda_k,-\mathcal{B}} = a - \lambda_k b$ until hitting the boundary. The critical value is determined by solving $a_i - \lambda b_i = \pm\lambda$. The solution gives the "hitting time" of coordinate i

$$t_i^{(\text{hit})} = \lambda = \frac{a_i}{b_i \pm 1} = \frac{[(D_{-\mathcal{B}}(D_{-\mathcal{B}})^T)^+ D_{-\mathcal{B}}\mathbf{y}]_i}{[(D_{-\mathcal{B}}(D_{-\mathcal{B}})^T)^+ D_{-\mathcal{B}}(D_{\mathcal{B}})^T s]_i \pm 1}. \tag{2.40}$$

And h_{k+1} is taken as the maximum of these hitting times

$$h_{k+1} = \max_i t_i^{(\text{hit})}. \tag{2.41}$$

For the leaving time, according to equations (2.37a)-(2.37e), we can obtain

$$t_i^{(\text{leave})} = \begin{cases} c_i/d_i, & c_i < 0, d_i < 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.42}$$

where c_i, d_i are defined as

$$c_i = s_i[D_{\mathcal{B}}[I - (D_{-\mathcal{B}})^{\intercal}(D_{-\mathcal{B}}(D_{-\mathcal{B}})^{\intercal})^+ D_{-\mathcal{B}}]\mathbf{y}]_i, \tag{2.43a}$$

$$d_i = s_i[D_{\mathcal{B}}[I - (D_{-\mathcal{B}})^{\intercal}(D_{-\mathcal{B}}(D_{-\mathcal{B}})^{\intercal})^+ D_{-\mathcal{B}}](D_{\mathcal{B}})^{\intercal} s]_i. \tag{2.43b}$$

And similarly, the next leaving time is

$$l_{k+1} = \max_i t_i^{(\text{leave})}. \tag{2.44}$$

Then the iteration will move until next hitting time or leaving time. The path algorithm is summarized in Algorithm 2.8.

Algorithm 2.8: Solution path algorithm for generalized Lasso

1 Initialize $k = 0, \lambda_0 = \infty, \mathcal{B} = \emptyset, and\ s = \emptyset$;
2 Compute the solution at λ_k by equation (2.39);
3 Compute the next hitting time h_{k+1} by (2.40) and (2.41) and next leaving time l_{k+1} by (2.42)-(2.44);
4 Set $\lambda_{k+1} = \max\{h_{k+1}, l_{k+1}\}$. If $h_{k+1} > l_{k+1}$, add the hitting coordinate to boundary list $\mathcal{B}$ and its sign s; otherwise remove the leaving coordinate from boundary $\mathcal{B}$ and its sign from s;
5 If $\lambda_k > 0$, Set $k = k + 1$, return to step 2; else, end the algorithm.

When X is a general design matrix in equation (2.33), the situation is more complicated. The dual problem is

$$\begin{aligned} \min_{\mathbf{u}} \quad & \frac{1}{2}(X^\intercal \mathbf{y} - D^\intercal \mathbf{u})^\intercal (X^\intercal X)^+ (X^\intercal \mathbf{y} - D^\intercal \mathbf{u}) \\ s.t. \quad & \|\mathbf{u}\|_\infty \le \lambda, \\ & D^\intercal \mathbf{u} \in \operatorname{row}(X). \end{aligned}$$

Tibshirani et al. [32] rewrite the above problem to be

$$\begin{aligned} \min_{\mathbf{u}} \quad & \frac{1}{2}\|\tilde{\mathbf{y}} - \tilde{D}^\intercal \mathbf{u}\|_2^2 \\ s.t. \quad & \|\mathbf{u}\|_\infty \le \lambda, \\ & D^\intercal \mathbf{u} \in \operatorname{row}(X), \end{aligned}$$

where $\tilde{\mathbf{y}} = XX^+\mathbf{y}, \tilde{D} = DX^+, X^+ = (X^\intercal X)^+ X^\intercal$. It is similar to equation (2.35), and the solution path can be obtained by the previous algorithm.

2.2.8 Solution path algorithm for weighted SVR

Karasuyama et al. [36] developed a solution path algorithm of weighted SVR (WSVR), which can deal with multiple hyperparameters. The optimization model of WSVR is

$$\min_{\beta_0,\boldsymbol{\beta},\xi^{(*)}} \quad \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \sum_{i=1}^{n} C_i(\xi_i + \xi_i^*) \tag{2.47a}$$

$$s.t. \quad y_i - (\langle \boldsymbol{\beta}, \phi(\mathbf{x}_i)\rangle + \beta_0) \le \epsilon + \xi_i \tag{2.47b}$$

$$(\langle \boldsymbol{\beta}, \phi(\mathbf{x}_i)\rangle + \beta_0) - y_i \le \epsilon + \xi_i^* \tag{2.47c}$$

$$\xi_i^{(*)} \ge 0, \quad i = 1, ..., n \tag{2.47d}$$

where the hyperparameter C_i defines the weight of how much ith instance inside the margin contribute to the overall error. The hyperparameter $\epsilon > 0$ defines the width of insensitive zone, where points inside the zone are given no penalty. The dual problem can be written as

$$\max_{\{\alpha_i\}_{i=1}^n} \quad -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j K(x_i, x_j) - \epsilon\sum_{i=1}^{n}|\alpha_i| + \sum_{i=1}^{n} y_i\alpha_i$$

$$s.t. \quad \sum_{i=1}^{n}\alpha_i = 0, -C_i \le \alpha_i \le C_i, i = 1, ..., n$$

Then the regression function can be written as $f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i K(\mathbf{x}, \mathbf{x}_i) + \beta_0$. The KKT conditions of the dual problem are:

$$\begin{aligned}
&|y_i - f(x_i)| \le \epsilon, \text{ if } \alpha_i = 0\\
&|y_i - f(x_i)| = \epsilon, \text{ if } 0 < |\alpha_i| < C_i\\
&|y_i - f(x_i)| \ge \epsilon, \text{ if } |\alpha_i| = C_i\\
&\sum_{i=1}^{n}\alpha_i = 0.
\end{aligned}$$

and the training points can be partitioned into three index sets: $\mathcal{O} = \{i : |y_i - f(\mathbf{x}_i)| \ge \epsilon, |\alpha_i| = C_i\}$, $\mathcal{E} = \{i : |y_i - f(\mathbf{x}_i)| = \epsilon, 0 < |\alpha_i| < C_i\}$, $\mathcal{I} = \{i : |y_i - f(\mathbf{x}_i)| \le \epsilon, \alpha_i = 0\}$.

Let

$$\mathbf{K}^{\mathcal{E}} = \begin{bmatrix} 0 & \mathbf{1}^{\intercal} \\ \mathbf{1} & \mathbf{K}_{\mathcal{E}} \end{bmatrix}, \mathbf{s} = \begin{bmatrix} sign(y_1 - f(\mathbf{x}_1)) \\ \vdots \\ sign(y_n - f(\mathbf{x}_n)) \end{bmatrix},$$

then from the KKT conditions, we can obtain the following linear relationship:

$$\begin{bmatrix} b \\ \boldsymbol{\alpha}_{\mathcal{E}} \end{bmatrix} = -(\mathbf{K}^{\mathcal{E}})^{-1} \begin{bmatrix} \mathbf{1}_{\mathcal{O}}^{\intercal} \\ \mathbf{K}_{\mathcal{E},\mathcal{O}} \end{bmatrix} \mathrm{diag}(s_{\mathcal{O}})\mathbf{c}_{\mathcal{O}} + (\mathbf{K}^{\mathcal{E}})^{-1} \begin{bmatrix} 0 \\ \mathbf{y}_{\mathcal{E}} - \epsilon \mathbf{s}_{\mathcal{E}} \end{bmatrix},$$
$$\boldsymbol{\alpha}_{\mathcal{O}} = \mathrm{diag}(\mathbf{s}_{\mathcal{O}})\mathbf{c}_{\mathcal{O}}, \quad \boldsymbol{\alpha}_{\mathcal{I}} = \mathbf{0},$$

where $\mathrm{diag}(\mathbf{s}_{\mathcal{O}})$ means the diagonal elements of $\mathbf{s}_{\mathcal{O}}$. Then the same method used in WSVC (Section 2.3.5) can be applied to WSVR to get the solution path with respect to $\mathbf{c}$.

Mao et al. [61] also proposed a solution path algorithm for WSVR. They extended the ϵ-path algorithm [35] to WSVR. Based on particle swarm optimization, they developed a heuristic weight-setting optimization algorithm. Mao et al. [62] also proposed a solution path algorithm for WSVR to deal with abnormal data.

2.2.9 Other solution path algorithm for regression

Solution path algorithm for Kernel Quantile Regression. Quantile Regression [63] can be used to estimate the quantiles of conditional density $p(y|\mathbf{x})$ as functions of $\mathbf{x}$. Kernel-based quantile regression (KQR) is proposed by Takeuchi et al. [64] and Li et al. [65]. Takeuchi et al. [66] designed a solution path algorithm for KQR. This approach can be used in nonparametric conditional density estimates, but it can only be applied to scalar output problem.

Bi-level path following algorithm for Kernel Quantile Regression. Rosset [67] developed a solution path algorithm for the bi-level quantile regression problem based on previous solution path algorithms [31, 37, 65]. This algorithm uses the possible locations of optima to efficiently find the solution surface with respect to λ and τ. This approach can also be extended to other problems, like SVR.

Solution path algorithm for SVOR. Ordinal regression has wide applications in many areas, such as flight delays forecasting [68]. Support vector ordinal regression (SVOR) is an important technique to solve ordinal regression problems [69]. Based on the fixed-margin principle and the sum-of-margin principle, Shashua and Levin [69] proposed two versions of SVOR. For the fixed-margin based SVOR, Chu et al. [70] developed an improved algorithm by keeping the partial order of the multiple parallel discrimination hyperplanes. For the sum-of-margin based SVOR, Gu et al. [71] proposed a modified version for incremental learning. Gu et al. [72] developed a regularization path algorithm to find the solution path with respect to regularization parameter. QR decomposition is used to handle the singularities.

Solution path algorithm for generalized linear model. Park et al. [73] proposed the solution path algorithm for ℓ_1-regularized generalized linear models (GLM). Different from LARS paths, the GLM paths are not piecewise linear. Predictor-corrector method was used to efficiently compute the solution path. Tay et al. [74] changed the ℓ_1 penalty to elastic net penalty and proposed an efficient algorithm to fit the entire regularization path.

2.3 Solution Path Algorithms for Classification

In classification problems, $\mathbf{x}_i \in \mathbb{R}^m$ is the input feature and $y_i \in \{+1, -1\}$ is categorical for binary classification ($y_i \in \{1, ..., k\}$ for k class classification problems). Classification models can be formulated as

$$\min_{\boldsymbol{\beta}, \beta_0} \sum_{i=1}^{n} L(y_i f(\mathbf{x}_i)) + \lambda J(\boldsymbol{\beta}),$$

where $f(\mathbf{x}_i) = \beta_0 + \boldsymbol{\beta}^\intercal \mathbf{x}_i$ is the discriminant function. λ is the regularization hyperparameter. The penalty term can take the ℓ_1-norm penalty, ℓ_2-norm penalty, elastic net penalty and so on. A commonly used loss function is hinge loss for SVM [60, 75] . In this case, the solution path of dual variables is piecewise linear with respect to regularization hyperparameter λ. Table 2.2 is the summary of solution path algorithm for classification models. The names shown in this table for classification are explained in the following part of this section.

Table 2.2: Solution Path Algorithm for Classification Models

Model	Formulation	Hyperparameter	Solution path algorithm	Reference
ℓ_2-SVM	Equation (2.51)	λ	Algorithm 2.9	[37]
ℓ_1-SVM	Equation (2.59)	λ	Algorithm 2.10	[41]
DrSVM	Equation (2.65)	λ_1, λ_2	Algorithm 2.11	[44]
ν-SVM	Equation (2.78)	ν	Algorithm 2.12	[42, 43]
WSVC	Equation (2.83)	C_i	Algorithm 2.13	[36]

2.3.1 Solution path algorithm for ℓ_2-SVM (SVMPath algorithm)

For the two-class classification, the ℓ_2-SVM is stated as

$$\min_{\boldsymbol{\beta},\beta_0} \quad \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 + \sum_{i=1}^{n} \xi_i \tag{2.51a}$$

$$s.t. \quad 1 - y_i f(\mathbf{x}_i) \leq \xi_i, \quad i = 1, \cdots, n \tag{2.51b}$$

$$\xi_i \geq 0, \quad i = 1, \cdots, n \tag{2.51c}$$

$$f(\mathbf{x}) = \beta_0 + \boldsymbol{\beta}^\intercal \mathbf{x} \tag{2.51d}$$

where λ is the regularization parameter. The choice of λ is very important, because it controls the trade-off between the regularization effect and the hinge loss. The KKT conditions can be written as

$$\boldsymbol{\beta} - \frac{1}{\lambda}\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \tag{2.52a}$$

$$\sum_{i=1}^{n} y_i \alpha_i = 0 \tag{2.52b}$$

$$\alpha_i + \gamma_i - 1 = 0 \tag{2.52c}$$

$$\alpha_i(1 - y_i f(\mathbf{x}_i) - \xi_i) = 0 \tag{2.52d}$$

$$y_i \xi_i = 0 \tag{2.52e}$$

Based on the KKT conditions, the data points can be partitioned into three sets: $\mathcal{E} = \{i : y_i f(\mathbf{x}_i) = 1, 0 \leq \alpha_i \leq 1\}$, $\mathcal{L} = \{i : y_i f(\mathbf{x}_i) < 1, \alpha_i = 1\}$, $\mathcal{L} = \{i : y_i f(\mathbf{x}_i) < 1, \alpha_i = 1\}$. For the initialization, When λ is very large, $\boldsymbol{\beta}$ will be zero, and the initial values of β_0 and the α_i depend on whether $n_- = n_+$ or not:

- When $n_- = n_+$, for a sufficiently large λ, all the $\alpha_i = 1$. The initial $\beta_0 \in [-1, 1]$– any value gives the same loss $\sum_{i=1}^{n} \xi_i = n_+ + n_-$.
- When $n_+ > n_-$, $\boldsymbol{\beta} = \mathbf{0}$, the optimal choice for β_0 is 1, and the loss is $\sum_{i=1}^{n} \xi_i = n_-$.

The linear SVM can be extended to nonlinear case using kernel methods. The discriminant function $f(\mathbf{x})$ can be written as: $f(\mathbf{x}_i) = \frac{1}{\lambda} \sum_{j=1}^{n} \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j) + \beta_0, i = 1, ..., n$, where $K(\mathbf{x}_i, \mathbf{x}_j)$ is kernel function. The three sets will not change until an event happens. The event can be either:

- The initial event, which means 2 or more points start at the elbow, with their initial values of $\alpha \in [0, 1]$;
- A point from $\mathcal{L}$ has just entered $\mathcal{E}$, with its value of α_i initially 1;
- A point from $\mathcal{R}$ has reentered $\mathcal{E}$, with its value of α_i initially 0;
- One or more points in $\mathcal{E}$ has left the set, to join either $\mathcal{R}$ or $\mathcal{L}$.

Suppose we are between ℓth event and $(\ell+1)$th event, $|\mathcal{E}_\ell| = m$, and let $\alpha_i^\ell, \beta_0^\ell$ and λ_ℓ be the values of these parameters at the point of entry. Denote $\alpha_0 = \lambda\beta_0$, and $\alpha_0^\ell = \lambda_\ell\beta_0^\ell$, then we have: $f(\mathbf{x}) = \frac{1}{\lambda}(\sum_{j=1}^n \alpha_j y_j K(\mathbf{x}, \mathbf{x}_j) + \alpha_0)$. For $\lambda_\ell > \lambda > \lambda_{\ell+1}$ we can obtain

$$f(x) = \frac{1}{\lambda}[\sum_{j\in\mathcal{E}_\ell}(\alpha_j - \alpha_j^\ell)y_jK(\mathbf{x}, \mathbf{x}_j) + (\alpha_0 - \alpha_0^\ell) + \lambda_\ell f^\ell(\mathbf{x})].$$

For the points in elbow, we have $y_i f(\mathbf{x}_i) = 1$, then

$$\frac{1}{\lambda}[\sum_{j\in\mathcal{E}_\ell}(\alpha_j - \alpha_j^\ell)y_iy_jK(\mathbf{x}_i, \mathbf{x}_j) + (\alpha_0 - \alpha_0^\ell)y_i + \lambda_\ell] = 1, \quad \forall i \in \mathcal{E}_\ell.$$

Denote $\delta_j = \alpha_j^\ell - \alpha_j$, and we can simplify it to be

$$\sum_{j\in\mathcal{E}_\ell}\delta_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \delta_0 y_i = \lambda_\ell - \lambda, \quad \forall i \in \mathcal{E}_\ell.$$

It can be further written as

$$\mathbf{K}_\ell^*\delta + \delta_0\mathbf{y}_\ell = (\lambda_\ell - \lambda)\mathbf{1}, \tag{2.54}$$

where $\mathbf{K}_\ell^*$ is $m \times m$ matrix with ijth entry $y_iy_jK(\mathbf{x}_i, \mathbf{x}_j)$ for i and j in $\mathcal{E}_\ell$, and $\mathbf{y}_\ell$ is the m vector with entries $y_i, i \in \mathcal{E}_\ell$. Since $\sum_{i=1}^n y_i\alpha_i = 0$, then we can obtain $\sum_{j\in\mathcal{E}_\ell} y_j\delta_j = 0$. It can be written as:

$$\mathbf{y}_\ell^T\delta = 0, \tag{2.55}$$

Define the following matrix

$$\mathbf{A}_\ell = \begin{pmatrix} 0 & \mathbf{y}_l^T \\ \mathbf{y}_\ell & \mathbf{K}_\ell^* \end{pmatrix}, \quad \delta^a = \begin{pmatrix} \delta_0 \\ \delta \end{pmatrix}, \text{ and } \mathbf{1}^a = \begin{pmatrix} 0 \\ \mathbf{1} \end{pmatrix},$$

Then equations (2.54) and (2.55) can be written as: $\mathbf{A}_\ell\delta^a = (\lambda_\ell - \lambda)\mathbf{1}^a$. If $\mathbf{A}_\ell$ has full rank, then we can introduce $\mathbf{b}^a = \mathbf{A}_\ell^{-1}\mathbf{1}^a$, and we can obtain the following linear relationship

$$\alpha_j = \alpha_j^\ell - (\lambda_\ell - \lambda)b_j, j \in \{0\} \cup \mathcal{E}_\ell. \tag{2.56}$$

For $\lambda_{\ell+1} < \lambda < \lambda_\ell$, we can obtain

$$f(\mathbf{x}) = \frac{\lambda_\ell}{\lambda}[f^\ell(\mathbf{x}) - h^\ell(\mathbf{x})] + h^\ell(\mathbf{x}), \tag{2.57}$$

where $h^\ell(\mathbf{x}) = \sum_{j \in \mathcal{E}_\ell} y_j b_j K(\mathbf{x}, \mathbf{x}_j) + b_0$. The paths continue until one of the following events occur:

- A point in the $\mathcal{E}$ moves into $\mathcal{L}$ or $\mathcal{R}$, which means the corresponding α_i for $i \in \mathcal{E}_\ell$ reaches a boundary (0 or 1). For each i, the value of λ for which this happens can be computed from (2.56)
- A point in $\mathcal{L}$ or $\mathcal{R}$ moves into $\mathcal{E}$, which means the point attains $y_i f(\mathbf{x}_i) = 1$. This occurs for point i at

$$\lambda = \lambda_\ell \Big(\frac{f^\ell(\mathbf{x}_i) - h^\ell(\mathbf{x}_i)}{y_i - h^\ell(\mathbf{x}_i)} \Big). \tag{2.58}$$

Based on these conditions, we can get the largest $\lambda < \lambda_\ell$ for which an event happens, get $\lambda_{\ell+1}$ and update the sets. The solution path algorithm for ℓ_2-SVM is shown in Algorithm 2.9.

Algorithm 2.9: Solution path algorithm for ℓ_2-SVM (SVMPath Algorithm)

1 Initialize λ, α, β_0, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{R}$;
2 Find the break point λ_ℓ where an event happens using (2.56) and (2.58);
3 Update $\mathcal{L}$, $\mathcal{E}$, $\mathcal{R}$, and update α_j and $f(x)$ by equation (2.56) and (2.57);
4 Find the break points using (2.56) and (2.58), and choose the largest $\lambda < \lambda_\ell$ to be $\lambda_{\ell+1}$. Then return to step 3;
5 End the algorithm if $\mathcal{L}$ is empty or $\lambda = 0$.

When the data points are separable, the algorithm is terminated when $\mathcal{L}$ becomes empty. When the data points are not separable, the algorithm is terminated when λ becomes zero.

The SVMpath algorithm uses the active set method to solve the quadratic programming (QP) problem associated with SVM [76]. The active-set methods offer high precision and incremental training for the SVM. But active-set methods must deal with the problem of potential singularities [77, 78]. The algorithm will be unstable when the data set contains duplicate data points, nearly duplicate data points and points that are linearly dependent in the kernel space.

ISVMP algorithm. To handle singularities in the SVMPath algorithm, Ong et al. [38] proposed an improved solution path algorithm for support vector machine (ISVMP). The ISVMP algorithm tracks the solution path in a multidimensional feasible space. The tracking process can be formulated into a linear programming problem, and they developed update formula for the linear programs.

Ridge-adding-based solution path algorithm. The ISVMP algorithm needs to solve additional linear programs to track the solution path. Dai et al. [39] developed a novel ridge-adding-based method to handle singularities, where a random ridge term is added to each data point. This algorithm is more efficient than the ISVMP algorithm. However, it may fail in some cases since its ability to handle singularity problem was with probability one. Changing the added ridges with another random seed may help alleviate this issue. However, when the active sets are randomly chosen, this method will not work.

Simple SVMPath algorithm. Sentelle et al. [40] proposed a simple version of the solution path algorithm to deal with semidefinite kernels. They enforce single active set changes at each iteration to avoid solving singular systems of equations. An artificial variable method is introduced to deal with unequal class cases. The simple SVMPath algorithm does not need to use external solver. Singular Value Decomposition and QR factorizations are also not required.

2.3.2 Solution path algorithm for ℓ_1-SVM

Different from ℓ_2-SVM, the ℓ_1-SVM achieves good performance when there are redundant noise features because the ℓ_1-norm penalty does a kind of continuous feature selection. The ℓ_1-SVM is formulated by replacing the ridge penalty in the ℓ_2-SVM with the Lasso penalty [41, 56]:

$$\min_{\boldsymbol{\beta},\beta_0} \sum_{i=1}^{n} [1 - y_i f(\mathbf{x}_i)]_+ + \lambda \|\boldsymbol{\beta}\|_1. \tag{2.59}$$

The above equation is equivalent to:

$$\begin{aligned} &\min_{\boldsymbol{\beta},\beta_0} \sum_{i=1}^{n} [1 - y_i f(\mathbf{x}_i)]_+ \\ &s.t.\ \|\boldsymbol{\beta}\|_1 = |\beta_1| + ... + |\beta_q| \leq s, \end{aligned}$$

where s is tuning parameter. The solution is denoted as $\hat{\beta}_0(s)$ and $\hat{\boldsymbol{\beta}}(s)$.

The Lasso penalty was first proposed for regression problems [56]. It has also been used in classification under the framework of SVMs. Because of the ℓ_1 nature of the penalty, when λ is sufficiently large (or s is sufficiently small), it will cause some of the coefficients to be exactly zero. Thus, the Lasso penalty does a kind of continuous feature selection.

Zhu et al. [41] proved the solution path $\hat{\boldsymbol{\beta}}(s)$ of the ℓ_1-SVM is piecewise linear, and developed an efficient algorithm to compute the whole solution path of ℓ_1-SVM.

Define $\mathcal{V} = \{j : \hat{\beta}_j(s) \neq 0\}, \mathcal{E} = \{i : 1 - y_i \hat{f}_i = 0\}, \mathcal{L} = \{i : 1 - y_i \hat{f}_i > 0\}$. As Δs increase, the sets will not change until an event happens: a training point moves into $\mathcal{E}$ or a basis function in $\mathcal{V}$ leaves $\mathcal{V}$. The current $\hat{\beta}_0, \hat{\boldsymbol{\beta}}$ and s are denoted by $\hat{\beta}_0^{\text{old}}, \hat{\boldsymbol{\beta}}^{\text{old}}$ and s^{old}. For each $j^* \notin \mathcal{V}$, we can obtain

$$u_0 + \sum_{\mathcal{V}} u_j h_j(\mathbf{x}_j) + u_{j^*} h_{j^*}(\mathbf{x}_i) = 0 \text{ for } i \in \mathcal{E}, \quad \sum_{\mathcal{V}} sign(\hat{\beta}_j^{\text{old}}) \cdot u_j + |u_{j^*}| = 1,$$

where u_0, u_j and u_{j^*} are unknowns. Then compute

$$\frac{\Delta loss_{j^*}}{\Delta s} = \sum_{\mathcal{L}} y_i(u_0 + \sum_{\mathcal{L}} u_j h_j(\mathbf{x}_j) + u_{j^*} h_{j^*}(\mathbf{x}_i)). \tag{2.62}$$

For each $i' \in \mathcal{E}$, we can obtain

$$u_0 + \sum_{\mathcal{V}} u_j h_j(\mathbf{x}_j) = 0 \text{ for } i \in \mathcal{E} \setminus \{i'\}, \quad \sum_{\mathcal{V}} sign(\hat{\beta}_j^{\text{old}}) \cdot u_j = 1,$$

where u_0 and u_j are unknowns. Then compute

$$\frac{\Delta loss_{i'}}{\Delta s} = \sum_{\mathcal{L}} y_i(u_0 + \sum_{\mathcal{L}} u_j h_j(\mathbf{x}_j)). \tag{2.64}$$

Then we can obtain the procedure of this algorithm shown in Algorithm 2.10.

Algorithm 2.10: Solution path algorithm for ℓ_1-SVM

---

1. Initialize $s = 0$, $\Delta s = 0$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$;
2. Increase Δs until an event happens, and compute $\frac{\Delta loss}{\Delta s}$ by equation (2.62) and (2.64);
3. Compare the computed values of $\frac{\Delta loss}{\Delta s}$ from above equations. Choose the smallest negative $\frac{\Delta loss}{\Delta s}$:
   - If the smallest $\frac{\Delta loss}{\Delta s}$ is non-negative, the algorithm terminates;
   - If the smallest $\frac{\Delta loss}{\Delta s}$ corresponds to a j^*, the algorithm update $\mathcal{V}$ by $\mathcal{V} \leftarrow \mathcal{V} \cup \{j^*\}, u \leftarrow (u, u_{j^*})^{\intercal}$;
   - If the smallest negative $\frac{\Delta loss}{\Delta s}$ corresponds to a i', the algorithm update $\mathcal{E}$ and $\mathcal{L}$ by $\mathcal{E} \leftarrow \mathcal{E} \setminus \{i'\}, \mathcal{L} \leftarrow \mathcal{L} \cup \setminus \{i'\}$ if necessary;

   In either of the last two cases, the solution $\hat{\beta}(s)$ updates: $\hat{\beta}_0(s^{old} + \Delta s) = \hat{\beta}_0^{old} + \Delta s \cdot u_0, \quad \hat{\beta}_{\mathcal{V}}(s^{old} + \Delta s) = \hat{\beta}_{\mathcal{V}}^{old} + \Delta s \cdot u$. Then go back to increasing Δs.

In the end, this algorithm gets a path $\hat{\beta}(s)$, which is piece-wise linear. It can efficiently compute the whole solution path $\hat{\beta}(t)$ of the ℓ_1-norm SVM, and facilitate adaptive selection of the tuning parameter s.

2.3.3 Solution path algorithm for DrSVM

ℓ_1-norm penalty can do automatic feature selection and has better performance than ℓ_2-norm penalty when there are redundant noise features. However, the ℓ_1-norm penalty also has some limitations [30]. When the variables are highly correlated, the ℓ_1-norm penalty tends to pick only a few of them and shrinks the rest to 0. For the high dimensional data ($p \gg n$), the ℓ_1-norm penalty can keep at most n input variables. Zou et al. [30] proposed the elastic net penalty to overcome these limitations. Wang et al. [44] apply the elastic net penalty to SVM, and obtain the doubly regularized support vector machine (DrSVM):

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n}[1 - y_i(\beta_0 + x_i^T\boldsymbol{\beta})]_+ + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1, \tag{2.65}$$

where λ_1 and λ_2 are tuning parameters for ℓ_1-norm penalty and ℓ_2-norm penalty. The elastic net penalty consists of the ℓ_1-norm penalty and the ℓ_2-norm penalty. The ℓ_1-norm penalty can do feature selection, and the ℓ_2 norm penalty has grouping effect for correlated variables.

Denote $\boldsymbol{\beta}_{\lambda_2}(\lambda_1)$ to be the solution path for a fixed value of λ_2, and similarly, $\boldsymbol{\beta}_{\lambda_1}(\lambda_2)$ is the solution path for a fixed value of λ_2. Wang et al. [44] proved that:

- When λ_2 is fixed, the solution $\boldsymbol{\beta}_{\lambda_2}(\lambda_1)$ is a piecewise linear function of λ_1;
- When λ_1 is fixed, the solution $\boldsymbol{\beta}_{\lambda_1}(\lambda_2)$ is a piecewise linear function of $1/\lambda_2$.

When λ_2 is fixed, then the equation (2.65) can be written as

$$\min_{\beta_0,\boldsymbol{\beta}} \quad \sum_{i=1}^{n}\xi_i + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 \tag{2.66a}$$

$$s.t. \quad 1 - y_i f(\mathbf{x}_i) \le \xi_i, \tag{2.66b}$$

$$\xi_i \ge 0, i = 1, ..., n \tag{2.66c}$$

$$\|\boldsymbol{\beta}\|_1 = |\beta_1| + \cdots + |\beta_p| \leq s \tag{2.66d}$$

And the KKT conditions are

$$\begin{aligned}
\lambda_2\beta_j - \sum_{i=1}^{n} \alpha_i y_i x_{ij} + \eta \cdot sign(\beta_j) &= 0, \quad j \in \mathcal{V} \\
\sum_{i=1}^{n} \alpha_i y_i &= 0, \\
1 - \alpha_i - \gamma_i &= 0, \quad i = 1, ..., n \\
\alpha_i(1 - y_i f(\mathbf{x}_i) - \xi_i) &= 0, \quad i = 1, ..., n \\
\gamma_i \xi_i &= 0, \quad i = 1, ..., n \\
\eta(\sum_{j=1}^{p} |\beta_j| - s) &= 0,
\end{aligned}$$

where $\alpha_i \geq 0, \gamma_i \leq 0, \eta \geq 0$ are Lagrange multipliers, and define the following sets: $\mathcal{E} = \{i : y_i f(\mathbf{x}_i) = 1, 0 \leq \alpha_i \leq 1\}$, $\mathcal{L} = \{i : y_i f(\mathbf{x}_i) < 1, \alpha_i = 1\}$, $\mathcal{R} = \{i : y_i f(\mathbf{x}_i) > 1, \alpha_i = 0\}$, $\mathcal{V} = \{j : \beta_j \neq 0\}$.

When s is small enough, constraint (2.66d) will be active. Then β_0 and $\boldsymbol{\beta}$ are determined by the following equations

$$\begin{aligned}
\lambda_2\beta_j - \sum_{i=1}^{n} \alpha_i y_i x_{ij} + \eta \cdot sign(\beta_j) &= 0, \quad j \in \mathcal{V} \\
\sum_{i=1}^{n} \alpha_i y_i &= 0, \\
y_i(\beta_0 + \sum_{j \in \mathcal{V}} \beta_j x_{ij}) &= 1, \quad i \in \mathcal{E} \\
\|\boldsymbol{\beta}\|_1 = \sum_{j \in \mathcal{V}} \beta_j \cdot sign(\beta_j) &= s,
\end{aligned}$$

If s increases are small enough, the sets $\mathcal{L}, \mathcal{R}, \mathcal{E}, \mathcal{V}$ will not change, so the above system will remain the same. By taking right derivatives, we can obtain

$$\lambda_2 \frac{\Delta\beta_j}{\Delta s} - \sum_{i \in \mathcal{E}} \frac{\Delta\alpha_i}{\Delta s} y_i x_{ij} + \frac{\Delta\eta}{\Delta s} \cdot sign(\beta_j) = 0, \quad j \in \mathcal{V}$$

$$\sum_{i\in\mathcal{E}} \frac{\Delta\alpha_i}{\Delta s} y_i = 0,$$

$$y_i(\frac{\Delta\beta_0}{\Delta s} + \sum_{j\in\mathcal{V}} \frac{\Delta\beta_j}{\Delta s} x_{ij}) = 1, \quad i \in \mathcal{E}$$

$$\|\boldsymbol{\beta}\|_1 = \sum_{j\in\mathcal{V}} \frac{\Delta\beta_j}{\Delta s} \cdot sign(\beta_j) = 1,$$

where the above system shows that the solution of $\alpha_i, \beta_j, \beta_0, \eta$ will change linearly in s until $\mathcal{L}, \mathcal{R}, \mathcal{E}, \mathcal{V}$ change. The following events are used to identify these changes:

- A point in $\mathcal{E}$ moves to $\mathcal{L}$ or $\mathcal{R}$, where α_i changes from within the $(0, 1)$ to the boundary 0 or 1. It will happen at

$$\delta_s^1 = \min_{i\in\mathcal{E}} \max(\frac{0-\alpha_i}{\Delta\alpha_i/\Delta s}, \frac{1-\alpha_i}{\Delta\alpha_i/\Delta s}). \tag{2.70}$$

- A point in $\mathcal{L}$ or $\mathcal{R}$ moves to $\mathcal{E}$, where $(1 - y_i f_i)$ reaches 0. It will happen at

$$\delta_s^2 = \min_{i\in\mathcal{E}_+^c}(\frac{0-r_i}{\Delta r_i/\Delta s}), \quad \mathcal{E}_+^c = \{i : \frac{0-r_i}{\Delta r_i/\Delta s} > 0, i \notin \mathcal{E}\}. \tag{2.71}$$

- An active point in $\mathcal{V}$ moves to becomes inactive, where a non-zero coefficient $\beta_j \neq 0$ becomes 0. It will happen at

$$\delta_s^3 = \min_{j\in\mathcal{V}_+}(\frac{0-\beta_j}{\Delta\beta_j/\Delta s}), \quad \mathcal{V}_+ = \{j : \frac{0-\beta_j}{\Delta\beta_j/\Delta s} > 0, j \in \mathcal{V}\}. \tag{2.72}$$

- An inactive point joins $\mathcal{V}$, where the absolute value of generalized correlation $(|c_j|)$ of variable j is equal to η. It will happen at

$$\delta_s^4 = \min_{j\notin\mathcal{V}_+} \max(\frac{-\eta - c_j}{\Delta c_j/\Delta s + \Delta\eta/\Delta s}, \frac{\eta - c_j}{\Delta c_j/\Delta s - \Delta\eta/\Delta s}), \tag{2.73}$$

  where

$$c_j = \lambda_2\beta_j - \sum_{i=1}^{n} \alpha_i y_i x_{ij}. \tag{2.74}$$

Generalized correlation of active variables reduces to zero will happen at

$$\delta_s^5 = \frac{0-\eta}{\Delta\eta/\Delta s}. \tag{2.75}$$

The main idea of the algorithm is to start λ_1 at a large number, then find the derivatives of β_0 and β_j with respect to λ_1. Next, the algorithm decreases λ_1 until an event happens and finds the new right derivatives.

For the initialization, $\boldsymbol{\beta}$ and s are initialized to be 0. There are two cases for initialization of β_0:

- When $n_+ = n_-$, any value of $\beta_0 \in [1, -1]$ will give the same loss;
- When $n_+ > n_-$, the optimal choice of β_0 is 1.

The solution path algorithm for DrSVM with fixed λ_2 is shown in Algorithm 2.11.

Similarly, when λ_1 is fixed, denote $D = 1/\lambda_2, \alpha_i^* = D\alpha_i$, and the KKT conditions become

$$\begin{aligned}\beta_j - \sum_{i=1}^{n} \alpha_i^* y_i x_{ij} &= -\lambda_1 D \cdot sign(\beta_j), \quad j \in \mathcal{V}\\ \sum_{i=1}^{n} \alpha_i^* y_i &= 0,\\ y_i(\beta_0 + \sum_{j\in\mathcal{V}} \beta_j x_{ij}) &= 1, \quad i \in \mathcal{E}\end{aligned}$$

By taking the right derivative of D, it can be proven that β_0 and $\boldsymbol{\beta}$ are piecewise linear in D. Similarly, α_i is piecewise linear in λ_2. Then we need to compute how much change of D (or λ_2) will lead to the next event:

- A point in $\mathcal{E}$ moves into $\mathcal{L}$ or $\mathcal{R}$: $\delta_{\lambda_2}^1 = \max_{i\in\mathcal{E}} \min(\frac{0-\alpha_i}{\Delta\alpha_i/\Delta\lambda_2}, \frac{1-\alpha_i}{\Delta\alpha_i/\Delta\lambda_2})$;
- A point in $\mathcal{L}$ or $\mathcal{R}$ moves into $\mathcal{E}$: $\delta_D^2 = \min_{i\in\mathcal{E}_+^c} \max(\frac{0-r_i}{\Delta r_i/\Delta D})$, where $\mathcal{E}_+^c = i : \frac{0-r_i}{\Delta r_i/\Delta D} > 0, i \notin \mathcal{E}$;

Algorithm 2.11: Solution path algorithm for DrSVM

1 Initialize s, Δs, β_0, $\boldsymbol{\beta}$, α,$\mathcal{R}$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$;

2 Compute the following derivatives

$$\frac{\Delta r_i}{\Delta s} = -y_i(\frac{\Delta\beta_0}{\Delta s} + \sum_{j\in\mathcal{V}} \frac{\Delta\beta_j}{\Delta s}), \quad i \notin \mathcal{E}$$

$$\frac{\Delta c_i}{\Delta s} = -\sum_{i\in\mathcal{V}} \frac{\Delta\alpha_i}{\Delta s} y_i x_{ij}, \quad j \notin \mathcal{V},$$

where r_i is the current residual $(1 - y_i f_i)$ for point i, and c_j is the current generalized correlation value for variable x_j;

3 Compute $\delta_s^1, \delta_s^2, \delta_s^3, \delta_s^4, \delta_s^5$ by equations (2.70)-(2.75), and find $\delta_s = \min(\delta_s^1, \delta_s^2, \delta_s^3, \delta_s^4, \delta_s^5)$;

4 Update $\alpha_i, \beta_0, \beta_j, \eta$, and update $\mathcal{L}, \mathcal{R}, \mathcal{E}$, and $\mathcal{V}$;

5 Stop the algorithm when one of the following situations happens:

- The generalized correlation reduces to zero;
- Two classes have been perfectly separated;
- A pre-specified maximum iteration number is reached.

Otherwise, compute the new derivatives, and go back to step 2.

- An active variable $(\beta_j \neq 0)$ becomes inactive: $\delta_D^3 = \min_{j\in\mathcal{V}_+}(\frac{0-\beta_j}{\Delta\beta_j/\Delta s})$, where $\mathcal{V}_+ = j : \frac{0-\beta_j}{\Delta\beta_j/\Delta D} > 0, j \in \mathcal{V}$;
- An inactive variable joins the active set: $\delta_{\lambda_2}^4 = \max_{j\notin\mathcal{V}_+} \min(\frac{-\lambda_1-c_j}{\Delta c_j/\Delta\lambda_2}, \frac{\lambda_1-c_j}{\Delta c_j/\Delta\lambda_2})$;
- λ_2 reduces to zero: $\delta_{\lambda_2}^5 = -\lambda_2$.

The event which happens first is determined by

$$\delta_{\lambda_2} = \max\left(\frac{-\lambda_2^2\delta_D^1}{1+\lambda_2\delta_D^1}, \delta_{\lambda_2}^2, \frac{-\lambda_2^2\delta_D^3}{1+\lambda_2\delta_D^3}, \delta_{\lambda_2}^4, \delta_{\lambda_2}^5\right).$$

Then the algorithm proceeds the same as fixed λ_2 situation.

DrSVM has better performance than ℓ_1-SVM and ℓ_2-SVM for high dimensional data. Variable selection is not bounded by the amount of training data, and it can select groups of correlated variables. The solution path algorithm for DrSVM highly facilitates selection of tuning parameters. Wang et al. [79, 80] replaced the hinge loss with huberized hinge loss in DrSVM, and proposed the hybrid huberized support vector machine (HHSVM). The HHSVM can select highly correlated variables together. An efficient solution path algorithm [79, 80] was proposed following an idea similar to DrSVM.

2.3.4 Solution path algorithm for ν-SVM

ν-support vector classification is proposed by Scholkopf et al. [81]. The formulation of ν-SVM is more complicated than C-SVM. Gu et al. [42] modified the formulation of ν-SVM and proposed an algorithm to track the regularization path based on two assumptions. In order to get rid of the two assumptions, Gu et al. [43] further proposed a new equivalent dual formulation of ν-SVM and developed a robust regularization path algorithm for ν-SVM. The optimization model of ν-SVC is

$$\min_{\boldsymbol{\beta},\beta_0,\xi_i} \quad \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 - v\rho + \frac{1}{n}\sum \xi_i \tag{2.78a}$$

$$s.t. \quad y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \quad i = 1, \cdots, n \tag{2.78b}$$

$$\xi_i \geq 0, \quad i = 1, \cdots, l, \quad \rho \geq 0 \tag{2.78c}$$

where the parameter ν is used to control the number of support vectors and margin errors. Denote $S^+ = \{(\mathbf{x}_i, y_i) \in S : y_i = +1\}$ and $S^- = \{(\mathbf{x}_i, y_i) \in S : y_i = -1\}$, the equivalent dual formulation can be written as

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^T Q \alpha \\ s.t. \quad & \sum_{i \in S^+} \alpha_i = \frac{\nu}{2}, \sum_{i \in S^-} \alpha_i = \frac{\nu}{2} \\ & 0 \leq \alpha_i \leq \frac{1}{l} \quad i = 1, \cdots, l \end{aligned}$$

Then we can obtain the KKT conditions

$$\forall i \in S : \tilde{g}_i = \frac{\partial W'_\nu}{\partial \alpha_i} = \sum_{j=1}^{l} \alpha_j Q_{ij} + I_{S^+}(i) d_1 + I_{S^-}(i) d_2,$$

$$\begin{cases} \tilde{g}_i \geq 0, & \text{for } \alpha_i = 0, \\ \tilde{g}_i = 0, & \text{for } 0 < \alpha_i < 1/l, \\ \tilde{g}_i \leq 0, & \text{for } \alpha_i = 1/l, \end{cases}$$

$$\frac{\partial W'_\nu}{\partial d_1} = \sum_{i \in S^+} \alpha_i = \frac{\nu}{2}, \quad \frac{\partial W'_\nu}{\partial d_2} = \sum_{i \in S^-} \alpha_i = \frac{\nu}{2},$$

where d_1, d_2 are Lagrangian multipliers. Based on the value of α_i, the training sample can be partitioned into S_S, S_E and S_R:

- $S_S = \{i : g_i = 0, 0 < \alpha_i < (1/l)\}$, and the set S_S includes margin support vectors strictly on the margins;
- $S_E = \{i : g_i \leq 0, \alpha_i = (1/l)\}$, and the set S_E includes error support vectors exceeding the margins;
- $S_R = \{i : g_i \geq 0, \alpha_i = 0\}$, and the set S_R includes the remaining vectors ignored by the margins.

In order to keep the KKT conditions satisfied, we can obtain the following linear system:

$$\Delta \tilde{g}_i = \sum_{j \in S_S} \Delta \alpha_j Q_{ij} + I_{S^+}(i) \Delta d_1 + I_{S^-}(i) \Delta d_2, \forall i \in S,$$

$$\sum_{i \in S^+} \Delta \alpha_i = \frac{\Delta \nu}{2}, \quad \sum_{i \in S^-} \Delta \alpha_i = \frac{\Delta \nu}{2}.$$

Let $[\tilde{\beta}_{d_1}, \tilde{\beta}_{d_2}, \tilde{\beta}_{S_S}^{\intercal}]$ denote the relationship between $\Delta \nu$ and $\Delta d_1, \Delta d_2, \Delta \alpha_{S_S}^{\intercal}$, then we can obtain

$$\underbrace{\begin{bmatrix} 0 & 0 & I_{S_S^+}(S_S)^{\intercal} \\ 0 & 0 & I_{S_S^-}(S_S)^{\intercal} \\ S_S^+(S_S) & S_S^-(S_S) & Q_{S_S S_S} \end{bmatrix}}_{\tilde{Q}} \begin{bmatrix} \tilde{\beta}_{d_1} \\ \tilde{\beta}_{d_2} \\ \tilde{\beta}_{S_S}^{T} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \mathbf{0} \end{bmatrix}.$$

If the samples are linearly dependent in RKHS, the $\tilde{Q}$ will be singular. Gu et al. [43] proposed a new method to avoid computing the inverse of $\tilde{Q}$, called ν-SvcRPath.

LU decomposition [82] is used in the ν-SvcRPath algorithm. Let L be a lower triangular matrix, U be an upper triangular matrix, and we can obtain: $P\tilde{Q} = LU$, where P is a permutation matrix. Forward substitution and back substitution [43,82] can be used to get the value of $\tilde{\beta}_{d_1}, \tilde{\beta}_{d_2}, \tilde{\beta}_{S_S}$. Then the relationship between Δg_i and $\Delta\xi$ can be obtained as

$$\Delta\tilde{g}_i = \left(\sum_{j\in S_S} \tilde{\beta}_j Q_{ij} + I_{S^+}(i)\tilde{\beta}_{d_1} + I_{S^-}(i)\tilde{\beta}_{d_2}\right)\Delta\nu = \tilde{\gamma}_i\Delta\nu.$$

The maximal adjustment quantity $\Delta\nu^{\max}$ is $\Delta\nu^{\max} = \min\{\Delta\nu^{S_S}, \Delta\nu^{S_R,S_E}, \nu\}$. Then we can update the variables α, g, d_1, d_2 as

$$\begin{aligned} \alpha_{S_S} &\leftarrow \alpha_{S_S} - \tilde{\beta}_{S_S}\Delta\nu^{\max}, \quad g \leftarrow g - \tilde{\gamma}\Delta\nu^{\max}, \\ d_1 &\leftarrow d_1 - \tilde{\beta}_{d_1}\Delta\nu^{\max}, \quad d_2 \leftarrow d_2 - \tilde{\beta}_{d_2}\Delta\nu^{\max}. \end{aligned}$$

And the sets S_S, S_E, S_R can be updated as:

- When point i is in S_S, if $\tilde{\beta}_i > 0$, point i will move from S_S to S_E; if $\tilde{\beta}_i < 0$, point i will move from S_S to S_R;
- When point i is in S_E, it will move from S_E to S_S;
- When point i is in S_R, it will move from S_R to S_S.

The algorithm of ν-SvcRPath is summarized in Algorithm 2.12.

2.3.5 Solution path algorithm for weighted SVC

The instance-weighted support vector machine [83–85], also called the weighted support vector machine (WSVM), is an extension of ordinary SVM, where each training

Algorithm 2.12: Solution path algorithm for ν-SVM

1. Initialize: $\nu \leftarrow \frac{2minn_+,n_-}{l}$;
2. Initialize $\alpha, \tilde{g}, d_1, d_2, S_S, S_E, S_R$;
3. Compute $\tilde{\beta}_{d_1}, \tilde{\beta}_{d_2}, \tilde{\beta}_{S_S}, \tilde{\gamma}, \Delta\nu^{max}$;
4. Update $\alpha, \tilde{g}, d_1, d_2, S_S, S_E, S_R$, and update $\nu \leftarrow \nu - \Delta\nu^{max}$;
5. If $\nu > 0$, return to step 3 until $\nu = 0$; else, end.

instance has its own weight. The optimization model is

$$\min_{\beta_0, \boldsymbol{\beta}} \quad \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + \sum_{i=1}^{n} C_i \xi_i \tag{2.83a}$$

$$s.t. \quad y_i(\boldsymbol{\beta}^\intercal \phi(\mathbf{x}_i) + \beta_0) \geq 1 - \xi_i, \quad i = 1, \cdots, n \tag{2.83b}$$

$$\xi_i \geq 0, \quad i = 1, \cdots, n \tag{2.83c}$$

where the hyperparameter C_i defines the weight of how much the ith instance inside the margin contributes to the overall error. When C_i is small, instances inside the margin are given a small penalty; when C_i is large, they are given a large penalty. If $C_i = C$ for all the instances, it reduces to the ordinary SVM. The dual problem of WSVM can be written as

$$\begin{aligned} \min_{\{\alpha_i\}_{i=1}^n} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j Q_{ij} - \sum_{i=1}^{n} \alpha_i \\ s.t. \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C_i, \quad i = 1, \cdots, n \end{aligned}$$

where $Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$, $K(\mathbf{x}_i, \mathbf{x}_j)$ is a reproducing kernel. Then the discriminant function can be written as: $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_j) + \beta_0$. Based on the KKT conditions, data points are partitioned into three sets:

$$\mathcal{O} = \{i : \alpha_i = 0\}, \quad \mathcal{M} = \{i : 0 < \alpha_i < C_i\}, \quad \mathcal{I} = \{i : \alpha_i = C_i\}.$$

Denote $\mathbf{c}_{\mathcal{I}}$ to be a sub-vector of $\mathbf{c}$ whose elements are indexed by $\mathcal{I}$. Similarly, $\mathbf{M}_{\mathcal{M},\mathcal{O}}$ means a sub-matrix of $\mathbf{M} \in \mathbb{R}^{n \times n}$ whose rows and columns are indexed by $\mathcal{M}$

and $\mathcal{O}$, and $\mathbf{M}_{\mathcal{M},\mathcal{M}}$ is abbreviated as $\mathbf{M}_{\mathcal{M}}$. Denote

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} Q_{11} & \cdots & Q_{1n} \\ \vdots & \ddots & \vdots \\ Q_{n1} & \cdots & Q_{nn} \end{bmatrix}, \quad \mathbf{M} = \begin{bmatrix} 0 & \mathbf{y}_{\mathcal{M}}^{\intercal} \\ \mathbf{y}_{\mathcal{M}} & \mathbf{Q}_{\mathcal{M}} \end{bmatrix}.$$

Assume $\mathbf{M}$ is invertible, then we can obtain:

$$\begin{bmatrix} b \\ \boldsymbol{\alpha}_{\mathcal{M}} \end{bmatrix} = -\mathbf{M}^{-1} \begin{bmatrix} \mathbf{y}_{\mathcal{I}}^{\intercal} \\ \mathbf{Q}_{\mathcal{M},\mathcal{I}} \end{bmatrix} \mathbf{c}_{\mathcal{I}} + \mathbf{M}^{-1} \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix}. \tag{2.86}$$

So b and $\boldsymbol{\alpha}_{\mathcal{M}}$ are affine with respect to $\mathbf{c}_{\mathcal{I}}$. For points in $\mathcal{I}$ and $\mathcal{O}$, we have

$$\boldsymbol{\alpha}_{\mathcal{I}} = \mathbf{c}_{\mathcal{I}}, \quad \boldsymbol{\alpha}_{\mathcal{O}} = \mathbf{0}. \tag{2.87}$$

An event means a change of the index sets $\mathcal{M}, \mathcal{O}, \mathcal{I}$. If no event happens, the WSVM solutions can be computed by equation (2.86) and equation (2.87).

Suppose the weight vector is updated from $\mathbf{c}^{(\text{old})}$ to $\mathbf{c}^{(\text{new})}$, then between the two events, we can obtain

$$\mathbf{c}(\theta) = \mathbf{c}^{(\text{old})} + \theta(\mathbf{c}^{(\text{new})} - \mathbf{c}^{(\text{old})}), \quad \theta \in [0, 1],$$

where θ is a parameter. The change of $\mathbf{c}$ can be written as: $\Delta\mathbf{c} = \Delta(\mathbf{c}^{(\text{new})} - \mathbf{c}^{(\text{old})}), \Delta\theta \geq 0$, where Δ represents the amounts of change from the current value. The following systems can be used to check whether the KKT conditions still hold:

$$\begin{aligned} y_i f(\mathbf{x}_i) + y_i \Delta f(\mathbf{x}_i) &\geq 1, \quad i \in \mathcal{O} \\ \alpha_i + \Delta\alpha_i &> 0, \quad i \in \mathcal{M} \\ \alpha_i + \Delta\alpha_i - (C_i + \Delta C_i) &< 0, \quad i \in \mathcal{M} \\ y_i f(\mathbf{x}_i) + y_i \Delta f(\mathbf{x}_i) &\leq 1, \quad i \in \mathcal{I} \end{aligned}$$

The above inequalities define the critical region, which is a convex polyhedron. The event points lie on the border of critical regions. Similar to $\mathbf{c}$, based on equation (2.86) the change of b and $\boldsymbol{\alpha}_{\mathcal{M}}$ can be written as

$$\begin{bmatrix} \Delta b \\ \Delta\boldsymbol{\alpha}_{\mathcal{M}} \end{bmatrix} = \Delta\theta\boldsymbol{\phi}, \tag{2.89}$$

where

$$\boldsymbol{\phi} = -\mathbf{M}^{-1} \begin{bmatrix} \mathbf{y}_{\mathcal{I}}^{\intercal} \\ \mathbf{Q}_{\mathcal{M},\mathcal{I}} \end{bmatrix} (\mathbf{c}^{(new)} - \mathbf{c}^{(old)}). \tag{2.90}$$

Then, $y_i \Delta f(\mathbf{x}_i)$ can be written as

$$\begin{aligned} y_i \Delta f(\mathbf{x}_i) &= \begin{bmatrix} y_i & \mathbf{Q}_{i,\mathcal{M}} \end{bmatrix} \begin{bmatrix} \Delta b \\ \Delta \boldsymbol{\alpha}_{\mathcal{M}} \end{bmatrix} + \mathbf{Q}_{i,\mathcal{I}} \Delta \mathbf{c}_{\mathcal{I}} \\ &= \Delta\theta \psi_i, \end{aligned}$$

where

$$\psi_i = \begin{bmatrix} y_i & \mathbf{Q}_{i,\mathcal{M}} \end{bmatrix} \boldsymbol{\phi} + \mathbf{Q}_{i,\mathcal{I}} (\mathbf{c}_{\mathcal{I}}^{(new)} - \mathbf{c}_{\mathcal{I}}^{(old)}). \tag{2.92}$$

Then the maximum step-length where no event happens can be obtained by

$$\Delta\theta = \min_{i \in \{1,\dots,|\mathcal{M}|,\}, j \in \mathcal{I} \cup \mathcal{O}} \left\{ -\frac{\alpha_{m_i}}{\phi_{i+1}}, \frac{C_{m_i} - \alpha_{m_i}}{\phi_{i+1} - d_{m_i}}, \frac{1 - y_j f(\mathbf{x}_j)}{\psi_j} \right\}_+, \tag{2.93}$$

where $d_i = C_i^{(\mathrm{new})} - C_i^{(\mathrm{old})}$, and the solution path can be computed by equation (2.89). At the border of critical region, the index sets $\mathcal{M}$, $\mathcal{O}$ and $\mathcal{I}$ are updated. Then the path proceeds again for the new critical region. The procedure is repeated until $\mathbf{c}$ reaches $\mathbf{c}^{(\mathrm{new})}$.

When $\mathcal{M}$ is empty, the KKT conditions can be rewritten as

$$\sum_{j \in \mathcal{I}} Q_{ij} C_j + y_i b \geq 1, \quad i \in \mathcal{O} \tag{2.94a}$$

$$\sum_{j \in \mathcal{I}} Q_{ij} C_j + y_i b \leq 1, \quad i \in \mathcal{I} \tag{2.94b}$$

$$\sum_{j \in \mathcal{I}} y_i C_i = 0 \tag{2.94c}$$

From the above conditions, we can obtain the range of optimal β_0

$$\max_{i \in \mathcal{L}} \; y_i g_i \leq \beta_0 \leq \min_{i \in \mathcal{U}} y_i g_i, \tag{2.95}$$

where $g_i = 1 - \sum_{j \in \mathcal{I}} Q_{ij} C_j, \mathcal{L} = \{i | i \in \mathcal{O}, y_i = 1\} \cup \{i | i \in \mathcal{I}, y_i = -1\}, \mathcal{U} = \{i | i \in \mathcal{O}, y_i = -1\} \cup \{i | i \in \mathcal{I}, y_i = 1\}$. Let $\delta \equiv \sum_{i \in \mathcal{I}} y_i d_i$, when $\delta = 0$, the step size $\Delta\theta$ can be increased as long as inequality (2.95) is satisfied. And $\Delta\theta$ is increased

$$u(\Delta\theta) = \max_{i\in\mathcal{U}} y_i(g_i + \Delta g_i(\Delta\theta)), \tag{2.96a}$$

$$\ell(\Delta\theta) = \min_{i\in\mathcal{L}} y_i(g_i + \Delta g_i(\Delta\theta)), \tag{2.96b}$$

where $\Delta g_i(\Delta\theta) = -\Delta \sum_{j\in\mathcal{I}} Q_{ij} d_j$. When $\delta \neq 0$, increasing $\Delta\theta$ will violate equation (2.94c). The following instance will enter $\mathcal{M}$:

$$i_{\text{low}} = \arg\max_{i\in\mathcal{L}} y_i g_i \quad \text{or} \quad i_{\text{up}} = \arg\min_{i\in\mathcal{U}} y_i g_i$$

And we can obtain the following updating rules:

$$\delta > 0 \Rightarrow \beta_0 = y_{i_{\text{up}}} g_{i_{\text{up}}}, \quad \mathcal{M} = \{i_{\text{up}}\} \tag{2.97a}$$

$$\delta < 0 \Rightarrow \beta_0 = y_{i_{\text{low}}} g_{i_{\text{low}}}, \quad \mathcal{M} = \{i_{\text{low}}\} \tag{2.97b}$$

The algorithm is shown in Algorithm 2.13.

Algorithm 2.13: Solution path algorithm for WSVM

1. Initialize $\theta = 0$, $\mathbf{c}^{(\text{old})}$, $\mathbf{c}^{(\text{new})}$, $\boldsymbol{\alpha}$, β_0, $\mathcal{M}$, $\mathcal{O}$, $\mathcal{I}$, Cholesky factor $\mathbf{L}$ of $\mathbf{Q}_{\mathcal{M}}$;
2. When $\mathcal{M}$ is empty, if $\delta(\boldsymbol{\alpha}) \neq 0$, set bias term b by (2.97); else, trace $u(\Delta\theta)$ and $\ell(\Delta\theta)$ in (2.96) until $u(\Delta\theta) = \ell(\Delta\theta)$;
3. When $\mathcal{M}$ is not empty, calculate $\boldsymbol{\phi}$, $\boldsymbol{\psi}$ and $\Delta\theta$ by (2.90), (2.92), (2.93);
4. If $\theta + \Delta\theta > 1$, then $\Delta\theta \leftarrow 1 - \theta$. Update $\boldsymbol{\alpha}, b, \mathbf{c}$ by step length $\Delta\theta$;
5. $\theta \leftarrow \theta + \Delta\theta$, Update $\mathcal{M}, \mathcal{O}, \mathcal{I}$ depending on the event type, Update $\mathbf{L}$;
6. If $\theta \neq 1$, return to step 2; else, end.

2.3.6 Other solution path algorithm for classification

Solution Path algorithm for Multicategory SVM. Inspired by the SVMPath algorithm for binary classification, Lee et al. [86] extended the path following algorithm to the Multicategory SVM (MSVM) [87]. They proved that the optimal value of coefficients is piecewise linear in $1/\lambda$ with an additional number of joints roughly proportional to the number of classes. Then, the entire solution path can be constructed sequentially.

ROC curve based solution path algorithm for SVM. Receiver Operating Characteristic (ROC) curves can used to measure the performance of classifier. It is the plot of true positive rate against false positive rate, and has a wide applications in machine learning [88, 89]. Batch et al. [90] extended the SVMPath algorithm to the case of asymmetric cost functions. Based on the piecewise linearity of the path of solutions, they developed a solution path algorithm for SVM to compute all possible ROC curves. The computational complexity is the same as training a single classifier.

Kernel path algorithm for SVM. Kernel methods [91] are widely applied in many machine learning problems. Wang et al. [92] developed an algorithm to compute the solution path with respect to kernel hyperparameter in SVC. An approximation method is developed to approximate the breakpoints because the kernel path is piecewise smooth instead of piecewise linear. This algorithm can also be applied to other kernel functions.

Solution path algorithm for OC-SVM. The one class SVM (OC-SVM) can be seen as the extension of support vector classification to the problem of one class classification [93, 94]. Lee and Scott [95] applied the SVMPath algorithm [37] to OC-SVM. Hastie et al. [37] have proven that the solution path of SVM is piecewise liner in λ. Thus, Lee and Scott reformulate the OC-SVM problem, and use λ to replace ν. In this way, they can compute the entire solution path for the OC-SVM.

Suboptimal solution path algorithm for SVM. Strict optimality is not necessary in some machine learning problems. Karasuyama and Takeuchi [96] introduced a suboptimal solution path algorithm for the SVM. The tolerance level can be used to control the trade-off between the accuracy of the solution and the computational cost. The suboptimal solutions can be seen as the solution of a perturbed problem from the original problem. This interpretation gives us new insights on the properties of suboptimal solutions.

Solution surface algorithm for CS-SVM. In many classification problems, the costs will vary for different types of mistakes. Cost sensitive learning [97] was proposed to deal with unequal misclassification costs, such as boundary movement [98], 2C-SVM [99], 2ν-SVM [100], and cost-sensitive hinge loss [101]. Gu et al. [102] developed a two dimensional solution surface algorithm to compute the solution path with respect to two regularization parameters. The bi-parameter space partition algorithm is used to compute a two-dimensional solution surface. Then the two-dimensional validation error surface for each cross-validation fold is calculated. Finally, the cross-validation error surface is obtained by superposing K validation error surfaces, and the global minimum cross-validation error of CS-SVM can be obtained. Morsier et al. [103] also proposed a solution path algorithm for CS-SVM. They applied this method to solve the Novelty Detection problem [104].

Solution path algorithm for Pin-SVM. C-SVM is sensitive to the feature noise around the boundary, and quantile distance can be used to solve this problem. Quantile value is closely related to the pinball loss [105–107]. The hinge loss is a special case of pinball loss when $\tau = 0$. Huang et al. [108] established an algorithm to compute the entire solution path with respect to τ. They also extended the τ value to negative case, which leads to a better accuracy.

Solution path algorithm for PinTSVM. The twin SVM (TSVM) was proposed by Jayadeva et al. [109]. The main idea is to find two nonparallel hyperplanes by solving two smaller-sized quadratic programming problems, which will make the learning speed faster [110]. Xu et al. [111] added the pinball loss to TSVM to make it more robust to feature noise around the decision boundary. Yang et al. [112] proposed a solution path approach for the PinTSVM. They proved that both the primal and dual solutions of the PinTSVM are piecewise linear with respect to parameters c and τ. The proposed method is much less computationally complex than the grid search method.

Solution path algorithm for ranking SVM. Ranking SVM [113] can be used to personalize the order of search results. Similar to SVM classification, the solution of ranking SVM is piecewise linear with respect to regularization parameter λ. Zapien et al. [114] established an algorithm to efficiently compute the entire regularization path.

2.4 Conclusions

In this chapter, we conduct a survey on solution path algorithms for regression and classification problems. The solution path algorithms take advantage of the piecewise linear property of the optimal values of coefficients. Once the breakpoints are obtained, the entire solutions can be computed by linear interpolation between breakpoints. We summarize the solution path algorithms for regression and classification models in Table 2.1 and Table 2.2. For each model, we analyze its mathematical formulations and derive the piecewise linear coefficient paths, with a summary of the algorithm procedure for each model. Some advantages and limitations of different models and solution path algorithms are also discussed. Hyperparameter tuning, which plays a vital role in machine learning, is usually time consuming. But solution path algorithms can greatly increase the speed of this process.

Chapter 3
Solution Path Algorithm for Distributionally Robust Regression

In this chapter, we propose a general distributionally robust regression model based on distributionally robust optimization theory. The proposed model has piecewise linear loss function and elastic net penalty term. This model can generalize many other regression models. We prove the piecewise linear property of the optimal solutions to this model, which enables us to develop a solution path algorithm for the hyperparameter tuning. Doubly regularized least absolute deviations (DrLAD) regression model is proposed based this framework. A solution path algorithm is developed to speed up the tuning of two hyperparameters in this model. Numerical experiments are implemented to validate the performance of this model and the computational efficiency of the solution path algorithm.

The remainder of this chapter is organized as follows. Chapter 3.1 introduces the background of this chapter. In Chapter 3.2, a general framework for regression problems is proposed which incorporates many regression models. In Chapter 3.3, the piecewise linear property of the solutions is explored. In Chapter 3.4, a solution path algorithm for hyperparameter tuning in DrLAD model is developed. In Chapter 3.5, numerical experiments are implemented to test the performance of the model and the solution path algorithm. Finally, Chapter 3.6 concludes this chapter.

3.1 Introduction

Regression is a technique to estimate the relationships between input $\mathbf{x} \in \mathbb{R}^d$ and output $y \in \mathbb{R}$. Regression coefficients $\boldsymbol{\beta}$ and β_0 (representing the relationships between input $\mathbf{x}$ and output y) can be estimated by solving expected risk minimization problem

$$\min_{\boldsymbol{\beta},\beta_0} \mathbb{E}^{\mathbb{P}}[\ell(z)], \tag{3.1}$$

where $\ell(z)$ is the loss function, $z = y - (\boldsymbol{\beta}^\intercal \mathbf{x} + \beta_0)$ is the residual. Here, $\mathbb{P}$ is the input-output distribution.

Different loss functions will lead to different regression models. The Ordinary Least Squares (OLS) takes the squared error as the loss function. But it is very sensitive to outlier points. The Least Absolute Deviation (LAD) [115] model minimizes the absolute error loss, which is more robust to outliers. Huber regression [116,117] models combine the squared error loss and absolute loss, where the loss function is squared error for residuals around zero and is absolute error for residuals far away from zero. But Huber loss introduces another hyperparameter to define the range for squared error and absolute error. Support vector regression [118] adopts the ϵ-insensitive loss to find the appropriate regression hyperplane, which is robust to outliers. Pinball loss is used in quantile regression [63], which can produce accurate quantile forecast.

In real applications, data will contain noise that makes it deviate from original values; this is called data uncertainty. With data uncertainty, the model will suffer from overfitting and have bad generalization performance in test data. Regularization is often used to handle overfitting. Hoerl et al. [119] added the ℓ_2-norm regularization to OLS model, which can achieve better prediction performance. The ℓ_1-norm regularization [56] can shrink the coefficient and do feature selection. The elastic net regularization was proposed by Zou and Hastie [30], and it encourages a grouping effect for correlated input features.

Instead of directly using the uniform empirical distribution over the training data, distributionally robust optimization (DRO) [120, 121] considers a family of distributions in an ambiguity set. Wasserstein metric [11, 15, 18] is a popular probability metric used to construct the ambiguity set. It contains all the distributions around the uniform empirical distribution of the training samples within a certain Wasserstein distance. The Wasserstein distance can be interpreted as the minimum transportation cost for moving unit mass of one distribution to the other distribution. The Wasserstein ambiguity set can include both discrete and continuous relevent distributions, and modern measure concentration results guarantee that the unknown data-generating distribution belongs to the set with high confidence for a sufficiently large sample size.

The Wasserstein distributionally robust optimization model can be reformulated as regularized machine learning model [8]. The choice of the regularization hyperparameter will highly influence the generalization performance of the model.

In some machine learning models, the optimal solutions of the coefficients will change piecewise linearly with respect to hyperparameters. Once the break points can be found sequentially, all the optimal solutions can be obtained efficiently. These kinds of algorithms are called solution path algorithms [45]. Solution path algorithm can be used to speed up the tuning of the regularization hyperparameters. The Least Angle Regression (LARS) algorithm was proposed by Efron et al. [29] to fit the Lasso model. Inspired by LARS, Zou et al. [30] proposed LARS-EN algorithm for elastic net regression model. Tibshirani et al. [32] further extended the solution path algorithm to generalized Lasso problems. Wang et al. [55] proposed the solution path algorithm for regularized least absolute deviations regression model. For support vector regression problem, λ-path [33], ϵ-path [34] and two-dimensional path algorithms [35] were developed to tune the hyperparameter. Takeuchi et al. [66] proposed a solution path algorithm for kernel quantile regression.

However, there is no well developed algorithm for the hyperparameter tuning of Wasserstein DRO regression model. This motivates the study of developing efficient algorithms for Wasserstein DRO regression model. This chapter proposes a general distributionally robust regression model with piecewise linear loss function and elastic net regularization term. The estimated regression coefficients in this model are piecewise linear with respect to the hyperparameters. Doubly Regularized Least Absolute Deviations (DrLAD) regression model is developed based on this general model, which is the combination of LAD loss and elastic net regularization term. A new solution path algorithm is developed for tuning of the two hyperparameters in this model.

3.2 Distributionally Robust Optimization for Regression Problems

3.2.1 Distributionally robust regression models with Wasserstein ambiguity set

In practice, the distribution $\mathbb{P}$ in (3.1) is uncertain and only indirectly observable through the training data. Based on distributionally robust optimization theory, we can construct an ambiguity set which contains the unknown distribution $\mathbb{P}$ with high confidence.

Let $\Xi = \mathbb{R}^d \times \mathbb{R}$, the Wasserstein distance between two distributions $\mathbb{Q}$ and $\mathbb{Q}'$ supported on Ξ is defined as:

$$W(\mathbb{Q}, \mathbb{Q}') = \inf_{\Pi} \Big\{ \int_{\Xi^2} d((\mathbf{x}, y), (\mathbf{x}', y'))\Pi((\mathbf{x}, y), (\mathbf{x}', y')) : \\ \Pi \text{ is a joint distribution of } \xi \text{ and } \xi' \text{ with marginals } \mathbb{Q} \text{ and } \mathbb{Q}' \Big\},$$

where $d((\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2)) = \|\mathbf{x}_1 - \mathbf{x}_2\| + \kappa|y_1 - y_2|$ is a metric on Ξ. It means the minimum cost of moving the distribution to another one. The parameter $\kappa > 0$

reflects the relative emphasis between feature uncertainty and label mismatch. The larger the κ is, the more confidence in labels. Given the training data $\{\hat{\mathbf{x}}_i, \hat{y}_i\}_{i=1}^n$, the empirical distribution is defined as $\hat{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^n \delta_{(\mathbf{x}_i, y_i)}$. Then we can construct an ambiguity set centered around $\hat{\mathbb{P}}$ as follows:

$$\mathcal{B}_\lambda(\hat{\mathbb{P}}) = \{\mathbb{Q} : W(\mathbb{Q}, \hat{\mathbb{P}}) \leq \lambda\},$$

where λ is radius of the Wasserstein ball. Thus, we obtain the following distributionally robust optimization model

$$\min_{\boldsymbol{\beta},\beta_0} h(\boldsymbol{\beta}) + \sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}})} \mathbb{E}^{\mathbb{Q}}[\ell(z)], \tag{3.2}$$

where $h(\boldsymbol{\beta})$ is a function of regression coefficients $\boldsymbol{\beta}$. If $\Xi = \mathbb{R}^{d+1}$ and $\ell(z)$ is Lipschitz continuous with respect to z, according to Theorem 4 in [8], if $\kappa = \infty$, then (3.2) can be reformulated as

$$\min_{\boldsymbol{\beta},\beta_0} h(\boldsymbol{\beta}) + \frac{1}{n}\sum_{i=1}^n \ell(z_i) + \lambda \cdot Lip(\ell)\|\boldsymbol{\beta}\|_* \tag{3.3}$$

where $*$ means the dual norm, $Lip(\ell)$ represents the Lipschitz modulus of ℓ.

3.2.2 Distributionally robust regression models with piecewise linear loss function

Piecewise linear loss functions are very common in regression model, such as ϵ-insensitive loss, pinball loss and LAD loss. The piecewise linear loss function can be defined as

$$\ell(z) = \max_{k\leq K}\{a_k z + b_k\}.$$

where K is the number of pieces. Assume $|a_k| \leq 1$, then $Lip(\ell) = 1$. If we let $\lambda = \lambda_1$, $* = 1$, $h(\boldsymbol{\beta}) = \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2$, and plug in the loss function, we can obtain

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{n}\sum_{i=1}^n \ell(z_i) + \lambda_1\|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 \tag{3.4}$$

If we take different loss functions, the distributionally robust regression model (3.4) can generalize many different regression models.

Support vector regression. If we take $\ell(z) = \max\{0, |z| - \epsilon\}, \epsilon \geq 0,\ \lambda_1 = 0$, $\lambda_2 = \lambda$, then we can obtain the following support vector regression model [118]

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{n}\sum_{i=1}^{n} \max\{0, |z_i| - \epsilon\} + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$$

Gunter et al. [33] proposed a solution path algorithm for the tuning of hyperparameter λ. A two-dimensional path algorithm was developed by Wang et al. [35] for hyperparmeter λ and ϵ.

Quantile regression. If we take $\ell(z) = \max\{-\tau z, (1-\tau)z\}$, $\tau \in [0,1]$, $\lambda_1 = 0$, $\lambda_2 = \lambda$, then we can obtain quantile regression model [64]

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{n}\sum_{i=1}^{n} \max\{-\tau z_i, (1-\tau)z_i\} + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$$

For the hyperparameter τ, Takeuchi et al. [66] proposed a solution path algorithm to efficiently find the optimal solutions with respect to different τ.

Doubly regularized Least absolute deviations regression (DrLAD). If we take $\ell(z) = |z|$, then we can obtain

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{n}\sum_{i=1}^{n} |z_i| + \lambda_1\|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 \tag{3.5}$$

In DrLAD model, the loss function is LAD loss, which is robust to noise. The elastic net regularization term can perform feature selection and encourages grouping effect for correlated input features.It has two hyperparameters λ_1 and λ_2. There is no solution path algorithm for DrLAD model. In this chapter, we propose an efficient solution path algorithm for DrLAD model.

3.3 Piecewise Linear Solution Path for Distributionally Robust Regression Model

In the distributionally robust regression model (3.4), the ℓ_1-norm regularization can be replaced by a linear constraint. Then we can obtain

$$\min_{\boldsymbol{\beta},\beta_0,\xi_i} \quad \frac{1}{n}\sum_{i=1}^{n}\xi_i + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 \tag{3.6a}$$

$$s.t. \; a_k z_i + b_k \leq \xi_i, \quad i = 1, \ldots, n, \quad k = 1, \ldots, K \tag{3.6b}$$

$$\sum_{j=1}^{d} \boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) \leq s \tag{3.6c}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, n \tag{3.6d}$$

For every λ_1, there exists a s such that model (3.4) and model (3.6) are equivalent. The proposed model (3.6) is a convex quadratic programming problem with convex objective function and linear constraints. There are two hyperparameters in the model: s and λ_2. The hyperparameters control the trade-off between the loss function and regularization term. The performance of the models is highly influenced by the choice the hyperparameters. If we directly perform cross-validation to identify a proper value for these hyperparameters, we need to solve the quadratic programming problems repeatedly, which leads to high computational cost.

In this chapter, we propose a solution path algorithm to efficiently calculate the optimal solutions with respect to different hyperparameters, which highly speeds up the process of cross-validation. In the following part of this section, we will explore the piecewise linear properties of the optimal solutions in model (3.6).

3.3.1 Piecewise linear solution path with respect to s

The distributionally robust regression model (3.6) has two hyperparameters: s for the ℓ_1 regularization term and λ_2 for the ℓ_2 regularization term. We first fix λ_2, and find the optimal solutions with respect to s.

Theorem 3.1. If λ_2 is fixed, the optimal solutions $(\boldsymbol{\beta}^*, \beta_0^*)$ of distributionally robust regression model (3.6) are piecewise linear with respect to s.

Proof. The Lagrangian function of (3.6) is

$$\begin{aligned} L(\boldsymbol{\beta}, \beta_0, \xi; s, \lambda_2) =& \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \sum_{k=1}^{K}\sum_{i=1}^{n}\eta_i^k(a_k z_i + b_k - \xi_i) \\ &+ \theta(\sum_{j=1}^{d}\boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) - s) - \sum_{i=1}^{n}\gamma_i\xi_i, \end{aligned}$$

where η_i^k, γ_i $(i = 1, \dots, n)$ and θ are Lagrangian multipliers, and $z_i = y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0)$. Define the following sets

$$\mathcal{R}_k = \{i : a_k z_i + b_k \neq 0\}, \quad \mathcal{E}_k = \{i : a_k z_i + b_k = 0\}, \quad \mathcal{V} = \{j : \boldsymbol{\beta}_j \neq 0\}$$

The KKT conditions are

$$\frac{\partial L}{\partial \boldsymbol{\beta}_j} = \lambda_2\boldsymbol{\beta}_j - \sum_{k=1}^{K}\sum_{i=1}^{n} a_k\eta_i^k\mathbf{x}_{ij} + \theta \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.8a}$$

$$\frac{\partial L}{\partial \beta_0} = -\sum_{k=1}^{K}\sum_{i=1}^{n} a_k\eta_i^k = 0 \tag{3.8b}$$

$$\frac{\partial L}{\partial \xi_i} = \frac{1}{n} - \sum_{k=1}^{K}\eta_i^k - \gamma_i = 0, \quad i = 1, \dots, n \tag{3.8c}$$

$$\eta_i^k(a_k z_i + b_k - \xi_i) = 0, \quad i = 1, \dots, n, \quad k = 1, \dots, K \tag{3.8d}$$

$$\theta(\sum_{j=1}^{d}\boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) - s) = 0 \tag{3.8e}$$

$$\gamma_i\xi_i = 0, \quad i = 1, \dots, n \tag{3.8f}$$

For the points in $\mathcal{R}_k$ and $\mathcal{E}_k$, we can obtain

- For $i \in \mathcal{R}_k$: $\xi_i \neq 0$, $\gamma_i = 0$, $\sum_{k=1}^{K} \eta_i^k = \frac{1}{n}$
- For $i \in \mathcal{E}_k$: $\xi_i = 0$, $\sum_{k=1}^{K} \eta_i^k \leq \frac{1}{n}$

For the points in the active sets $\mathcal{V}, \mathcal{E}_k$, the corresponding constraints are active. The we can obtain the following system of equations

$$\lambda_2 \boldsymbol{\beta}_j - \sum_{k=1}^{K}\sum_{i=1}^{n} a_k \eta_i^k \mathbf{x}_{ij} + \theta \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.9a}$$

$$\sum_{k=1}^{K}\sum_{i=1}^{n} a_k \eta_i^k = 0, \tag{3.9b}$$

$$a_k(y_i - (\sum_{j\in\mathcal{V}} \boldsymbol{\beta}_j \mathbf{x}_{ij} + \beta_0)) + b_k = 0, \quad i \in \mathcal{E}_k, \quad k = 1, \dots, K \tag{3.9c}$$

$$\sum_{j\in\mathcal{V}} \boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) = s, \text{ if } \theta > 0 \tag{3.9d}$$

In the above linear system, the number of equations and number of unknowns are equal. When s increases by a small enough amount, the sets $\mathcal{R}_k$, $\mathcal{E}_k$, $\mathcal{V}$ will not change, and the structure of the above system will not change. Taking derivatives with respect to s, we have

$$\lambda_2 \frac{\Delta\boldsymbol{\beta}_j}{\Delta s} - \sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} \frac{\Delta\eta_i^k}{\Delta s} a_k \mathbf{x}_{ij} + \frac{\Delta\theta}{\Delta s} \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V}$$

$$\sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} a_k \frac{\Delta\eta_i^k}{\Delta s} = 0,$$

$$a_k(\sum_{j\in\mathcal{V}} \frac{\Delta\boldsymbol{\beta}_j}{\Delta s}\mathbf{x}_{ij} + \frac{\Delta\beta_0}{\Delta s}) = 0, \quad i \in \mathcal{E}_k, \quad k = 1, \dots, K$$

$$\sum_{j\in\mathcal{V}} \frac{\Delta\boldsymbol{\beta}_j}{\Delta s} \cdot sign(\boldsymbol{\beta}_j) = 1, \text{ if } \theta > 0$$

The solutions of the above system of linear equations are the gradients of $\boldsymbol{\beta}_j, \beta_0, \eta_i^k, \theta$. If the active sets $\mathcal{V}$ and $\mathcal{E}_k$ do not change, the gradients will not change. So $\boldsymbol{\beta}_j, \beta_0, \eta_i^k, \theta$ will change linearly in s. When the s increases large enough, one of the active set $\mathcal{V}$ and $\mathcal{E}$ will change, and the structure of the above system will change. We will obtain new gradients, so the optimal solutions will change piecewise linear in s. $\square$

3.3.2 Piecewise linear solution path with respect to λ_2

Next, we fix s, and find the optimal solutions of $\boldsymbol{\beta}$ and β_0 with respect to λ_2.

Theorem 3.2. If s is fixed, the optimal solutions $(\boldsymbol{\beta}^*, \beta_0^*)$ of distributionally robust regression model (3.6) are piecewise linear with respect to $1/\lambda_2$.

Proof. Similar to (3.9), let $\nu = 1/\lambda_2$, $\zeta_i^k = \nu\eta_i^k$, we have

$$\boldsymbol{\beta}_j - \sum_{k=1}^{K}\sum_{i=1}^{n} a_k\zeta_i^k \mathbf{x}_{ij} + \theta\nu \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.11a}$$

$$\sum_{k=1}^{K}\sum_{i=1}^{n} \zeta_i^k = 0, \tag{3.11b}$$

$$a_k(y_i - (\sum_{j\in\mathcal{V}} \boldsymbol{\beta}_j \mathbf{x}_{ij} + \beta_0)) + b_k = 0, \quad i \in \mathcal{E}_k, \quad k = 1, \dots, K \tag{3.11c}$$

In the above linear system, the number of equations and unknowns ($\theta = \lambda_1$ is known) are equal. When ν changes by a small enough amount, the sets $\mathcal{E}_k$, $\mathcal{V}$ will not change, and the structure of the above system will not change. The system of linear equations (3.11) can be written as

$$\boldsymbol{\beta}_j - \sum_{k=1}^{K}\sum_{i\in\mathcal{E}} a_k\zeta_i^k \mathbf{x}_{ij} - \sum_{k=1}^{K}\sum_{i\notin\mathcal{E}} \nu a_k\eta_i^k \mathbf{x}_{ij} + \theta\nu \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.12a}$$

$$\sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} a_k\zeta_i^k + \sum_{k=1}^{K}\sum_{i\notin\mathcal{E}} a_k\zeta_i^k = 0, \tag{3.12b}$$

$$a_k(y_i - (\sum_{j\in\mathcal{V}} \boldsymbol{\beta}_j \mathbf{x}_{ij} + \beta_0)) + b_k = 0, \quad i \in \mathcal{E}_k, \quad k = 1, \dots, K \tag{3.12c}$$

Taking derivatives with respect to ν, we can obtain

$$\frac{\Delta\boldsymbol{\beta}_j}{\Delta\nu} - \sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} \frac{\Delta\zeta_i^k}{\Delta\nu} a_k \mathbf{x}_{ij} = \sum_{k=1}^{K}\sum_{i\notin\mathcal{E}_k} a_k\eta_i^k \mathbf{x}_{ij} - \theta \cdot sign(\boldsymbol{\beta}_j), \quad j \in \mathcal{V}$$

$$\sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} a_k \frac{\Delta\zeta_i^k}{\Delta\nu} = -\sum_{k=1}^{K}\sum_{i\notin\mathcal{E}_k} a_k\eta_i^k = \sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} a_k\eta_i^k$$

$$a_k(\sum_{j\in\mathcal{V}} \frac{\Delta\boldsymbol{\beta}_j}{\Delta\nu}\mathbf{x}_{ij} + \frac{\Delta\beta_0}{\Delta\nu}) = 0, \quad i \in \mathcal{E}_k, \quad k = 1,\dots,K$$

Similar as the proof for Theorem 3.1, we can find $\boldsymbol{\beta}$, β_0 are piecewise linear in ν. In equation (3.11), if we plug in $\boldsymbol{\beta}_j = \frac{1}{\lambda_2}\sum_{k=1}^{K}\sum_{i=1}^{n} a_k\eta_i^k\mathbf{x}_{ij} - \frac{\theta}{\lambda_2}\cdot sign(\boldsymbol{\beta}_j)$

$$\sum_{k=1}^{K}\sum_{i=1}^{n} a_k\eta_i^k = 0,$$

$$a_k(y_i\lambda_2 - (\sum_{j\in\mathcal{V}}\sum_{k=1}^{K}(\sum_{i=1}^{n} a_k\eta_i^k\mathbf{x}_{ij} - \theta\cdot sign(\boldsymbol{\beta}_j))\mathbf{x}_{ij} + \eta_0)) + \lambda_2 b_k = 0, \quad i \in \mathcal{E}_k, \quad k = 1,\dots,K$$

where $\eta_0 = \lambda_2\beta_0$. Then taking derivative with respect to λ_2, we can obtain

$$\sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k} a_k\frac{\Delta\eta_i^k}{\Delta\lambda_2} = 0,$$

$$a_k(\sum_{j\in\mathcal{V}}(\sum_{k=1}^{K}\sum_{i\in\mathcal{E}_k}\frac{\Delta\eta_i^k}{\Delta\lambda_2}a_k\mathbf{x}_{ij})\mathbf{x}_{ij} + \frac{\Delta\eta_0}{\Delta\lambda_2}) = y_i + b_k, \quad i \in \mathcal{E}_k, \quad k = 1,\dots,K$$

Similar as the proof for Theorem 3.1, we can find that η_i^k is piecewise linear in λ_2. □

3.4 Solution Path Algorithm for Doubly Regularized Least Absolute Deviations Regression

In this section, we consider a special case of the distributionally robust regression model, the DrLAD model (shown in (3.5)). Similar as the reformulation of model (3.6), model (3.5) can be equivalently written as

$$\min_{\boldsymbol{\beta},\beta_0,\xi} \frac{1}{n}\sum_{i=1}^{n}\xi_i + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 \tag{3.16a}$$

$$s.t.\ y_i - (\boldsymbol{\beta}^\intercal\mathbf{x}_i + \beta_0) \le \xi_i, \quad i = 1,\dots,n \tag{3.16b}$$

$$(\boldsymbol{\beta}^\intercal\mathbf{x}_i + \beta_0) - y_i \le \xi_i, \quad i = 1,\dots,n \tag{3.16c}$$

$$\sum_{j=1}^{d}\boldsymbol{\beta}_j\cdot sign(\boldsymbol{\beta}_j) \le s \tag{3.16d}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, n \tag{3.16e}$$

In this model, we have two hyperparameters s and λ_2. A solution path algorithm is developed for this model.

3.4.1 Piecewise linear solution path with respect to s

The Lagrangian function of (3.16) is

$$\begin{aligned} L(\boldsymbol{\beta}, \beta_0, \xi; s, \lambda_2) =& \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\eta_i^+(y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) - \xi_i) \\ & - \sum_{i=1}^{n}\eta_i^-(y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) + \xi_i) + \theta(\sum_{j=1}^{d}\boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) - s) - \sum_{i=1}^{n}\gamma_i\xi_i \end{aligned}$$

where η_i^+, η_i^-, γ_i $(i = 1, \ldots, n)$ and θ are Lagrangian multipliers. Define $\mathcal{V} = \{j : \boldsymbol{\beta}_j \neq 0\}$, the KKT conditions are

$$\frac{\partial L}{\partial \boldsymbol{\beta}_j} = \lambda_2\boldsymbol{\beta}_j - \sum_{i=1}^{n}(\eta_i^+ - \eta_i^-)\mathbf{x}_{ij} + \theta \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.18a}$$

$$\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^{n}(\eta_i^+ - \eta_i^-) = 0 \tag{3.18b}$$

$$\frac{\partial L}{\partial \xi_i} = \frac{1}{n} - \eta_i^+ - \eta_i^- - \gamma_i = 0, \quad i = 1, \ldots, n \tag{3.18c}$$

$$\eta_i^+(y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) - \xi_i) = 0, \quad i = 1, \ldots, n \tag{3.18d}$$

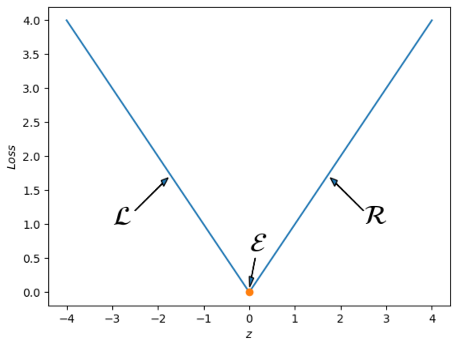
$$\eta_i^-(y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) + \xi_i) = 0, \quad i = 1, \ldots, n \tag{3.18e}$$

$$\theta(\sum_{j=1}^{d}\boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) - s) = 0 \tag{3.18f}$$

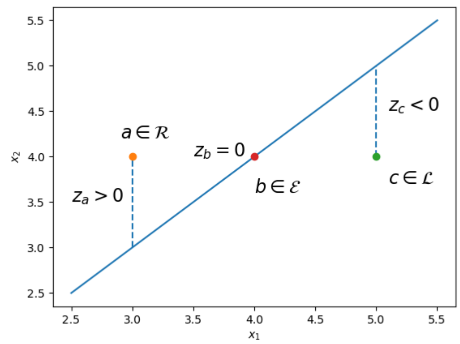
$$\gamma_i\xi_i = 0, \quad i = 1, \ldots, n \tag{3.18g}$$

Then, as shown in Figure 3.1, we can split the data points into different groups based on the residuals

$$\mathcal{E} = \{i : y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) = 0\}, \quad \mathcal{L} = \{i : y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) < 0\},$$



(a)



(b)

Figure 3.1: Partition of Data Points.

$$\mathcal{R} = \{i : y_i - (\boldsymbol{\beta}^\intercal \mathbf{x}_i + \beta_0) > 0\}$$

where $\mathcal{E}$ contains points in the elbow of the LAD loss. $\mathcal{L}$ corresponds to points in the left of the elbow, and $\mathcal{R}$ means points lie in the right of the elbow. $\mathcal{V}$ contains the index of the non zero feature. Then we have

- For $i \in \mathcal{R}$: $\xi_i > 0$, $\gamma_i = 0$, $\eta_i^+ = \frac{1}{n}$, $\eta_i^- = 0$
- For $i \in \mathcal{L}$: $\xi_i > 0$, $\gamma_i = 0$, $\eta_i^+ = 0$, $\eta_i^- = \frac{1}{n}$
- For $i \in \mathcal{E}$: $\xi_i = 0$, $\eta_i^+ + \eta_i^- \leq \frac{1}{n}$

Let $\eta_i = \eta_i^+ - \eta_i^-$, then we have the following equations

$$\lambda_2 \boldsymbol{\beta}_j - \sum_{i=1}^{n} \eta_i \mathbf{x}_{ij} + \theta \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.19a}$$

$$\sum_{i=1}^{n} \eta_i = 0, \tag{3.19b}$$

$$y_i - (\sum_{j \in \mathcal{V}} \boldsymbol{\beta}_j \mathbf{x}_{ij} + \beta_0) = 0, \quad i \in \mathcal{E} \tag{3.19c}$$

$$\sum_{j \in \mathcal{V}} \boldsymbol{\beta}_j \cdot sign(\boldsymbol{\beta}_j) = s, \text{ if } \theta > 0 \tag{3.19d}$$

This linear system consists of $|\mathcal{V}| + |\mathcal{E}| + 2$ equations and $|\mathcal{V}| + |\mathcal{E}| + 2$ unknowns. When s increases by a small enough amount, the sets $\mathcal{R}$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$ will not change, and the structure of the above system will not change. Taking right derivatives with respect to s, we have

$$\begin{aligned}\lambda_2\frac{\Delta\boldsymbol{\beta}_j}{\Delta s} - \sum_{i\in\mathcal{E}}\frac{\Delta\eta_i}{\Delta s}\mathbf{x}_{ij} + \frac{\Delta\theta}{\Delta s}\cdot sign(\boldsymbol{\beta}_j) &= 0, \quad j \in \mathcal{V}\\ \sum_{i\in\mathcal{E}}\frac{\Delta\eta_i}{\Delta s} &= 0,\\ \sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{\beta}_j}{\Delta s}\mathbf{x}_{ij} + \frac{\Delta\beta_0}{\Delta s} &= 0, \quad i \in \mathcal{E}\\ \sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{\beta}_j}{\Delta s}\cdot sign(\boldsymbol{\beta}_j) &= 1, \text{ if } \theta > 0\end{aligned}$$

We define the following events to identify changes in $\mathcal{V}$, $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{E}$.

- An active feature in $\mathcal{V}$ becomes inactive. This event happens when $|\boldsymbol{\beta}_j|$ becomes 0.

- An inactive feature joins active set $\mathcal{V}$. This event happens when the absolute value of the generalized correlation $c_j = \lambda_2\boldsymbol{\beta}_j - \sum_{i=1}^{n}\eta_i\mathbf{x}_{ij}$ reaches θ.

- A point moves from $\mathcal{E}$ to $\mathcal{L}$ or $\mathcal{R}$. This event happens when η_i becomes $-\frac{1}{n}$ or $\frac{1}{n}$.

- A point moves from $\mathcal{L}$ or $\mathcal{R}$ to $\mathcal{E}$. This event happens when the residual $r_i = y_i - (\boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i + \beta_0)$ reaches 0.

At the initial state, $s = 0$, $\boldsymbol{\beta} = 0$, the objective function of (3.5) becomes

$$\min_{\beta_0} \frac{1}{n}\sum_{i=1}^{n}|y_i - \beta_0|.$$

We have $\mathcal{V} = \emptyset$. Let $y_{(1)}, y_{(2)}, \ldots, y_{(n)}$ be the response variable in ascending order. If n is odd, let t_0 be the index corresponding to $y_{([n/2])}$, then we have $b = y_{t_0}$ and

$\eta_{t_0} = 0$. If n is even, let t_1 and t_2 be the corresponding indices of $y_{(n/2)}$ and $y_{(n/2+1)}$, then either one of them can be added to $\mathcal{E}$. If we take $b = y_{t_1}$, then we have $\eta_{t_1} = -\frac{1}{n}$.

Let δ^i be the smallest decrease for the first event i to happen. Next, we will detail exactly how to find each δ^i.

(1) An active feature in $\mathcal{V}$ becomes inactive:

$$\delta_s^1 = \min_{j \in \mathcal{V}_+} \left\{ \frac{0 - \boldsymbol{\beta}_j}{\Delta \boldsymbol{\beta}_j / \Delta s} \right\}, \tag{3.21}$$

where $\mathcal{V}_+ = \{j \in \mathcal{V} : \frac{0-\boldsymbol{\beta}_j}{\Delta\boldsymbol{\beta}_j/\Delta s} > 0\}$.

(2) An inactive feature joins active set $\mathcal{V}$. Let $f_j^+ = \frac{-\theta - c_j}{\Delta c_j/\Delta s + \Delta\theta/\Delta s}$, $f_j^- = \frac{\theta - c_j}{\Delta c_j/\Delta s - \Delta\theta/\Delta s}$, then we have

$$g_j = \begin{cases} \min\{f_j^+, f_j^-\}, \text{ if } f_j^+ > 0 \text{ and } f_j^- > 0 \\ \max\{f_j^+, f_j^-\}, \text{ if either } f_j^+ \leq 0 \text{ or } f_j^- \leq 0 \text{ but not both} \\ \infty, \text{ if } f_j^+ \leq 0 \text{ and } f_j^- \leq 0 \end{cases}$$

$$\delta_s^2 = \min_{j \notin \mathcal{V}} \{g_j\} \tag{3.22}$$

where $\frac{\Delta c_j}{\Delta s} = -\sum_{i \in \mathcal{E}} \frac{\Delta \eta_i}{\Delta s} \mathbf{x}_{ij}$

(3) A point moves from $\mathcal{E}$ to $\mathcal{L}$ or $\mathcal{R}$:

$$\delta_s^3 = \min_{i \in \mathcal{E}} \max \left\{ \frac{-\frac{1}{n} - \eta_i}{\Delta \eta_i / \Delta s}, \frac{\frac{1}{n} - \eta_i}{\Delta \eta_i / \Delta s} \right\} \tag{3.23}$$

(4) A point moves from $\mathcal{L}$ or $\mathcal{R}$ to $\mathcal{E}$:

$$\delta_s^4 = \min_{i \in \mathcal{E}_+^c} \left\{ \frac{0 - r_i}{\Delta r_i / \Delta s} \right\}, \tag{3.24}$$

where $\mathcal{E}_+^c = \{i \notin \mathcal{E} : \frac{0-r_i}{\Delta r_i/\Delta s} > 0\}$, $\frac{\Delta r_i}{\Delta s} = -\sum_{j \in \mathcal{V}} \frac{\Delta \boldsymbol{\beta}_j}{\Delta s} \mathbf{x}_{ij} - \frac{\Delta \beta_0}{\Delta s}$.

(5) the generalized correlation of active features reaches to 0:

$$\delta_s^5 = \frac{0 - \theta}{\Delta \theta / \Delta s} \tag{3.25}$$

The algorithm for the s-path is shown in Algorithm 3.1.

Algorithm 3.1: s-path algorithm

1 Initialize s, λ_2, β_0, $\boldsymbol{\beta}$, η_i, θ, $\mathcal{R}$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$;

2 Compute $\delta_s^1, \delta_s^2, \delta_s^3, \delta_s^4, \delta_s^5$ by equations (3.21)-(3.25), and find $\delta_s = \min(\delta_s^1, \delta_s^2, \delta_s^3, \delta_s^4, \delta_s^5)$;

3 Update $\boldsymbol{\beta}_j, \beta_0, \eta_i, \theta$ by

$$\boldsymbol{\beta}_j \leftarrow \boldsymbol{\beta}_j + \delta_s \frac{\Delta \boldsymbol{\beta}_j}{\Delta s}, \quad j \in \mathcal{V}, \qquad \beta_0 \leftarrow \beta_0 + \delta_s \frac{\Delta \beta_0}{\Delta s}$$
$$\eta_i \leftarrow \eta_i + \delta_s \frac{\Delta \eta_i}{\Delta s}, \quad i \in \mathcal{E}, \qquad \theta \leftarrow \theta + \delta_s \frac{\Delta \theta}{\Delta s}$$

Update $\mathcal{L}, \mathcal{R}, \mathcal{E}$, and $\mathcal{V}$;

4 Set $s \leftarrow s + \delta_s$;

5 Stop the algorithm when one of the following situation happens:

- The generalized correlation reduces to zero;
- s reaches to upper bound.

Otherwise, return to step 2.

3.4.2 Piecewise linear solution path with respect to λ_2

Similar as (3.19), let $\nu = 1/\lambda_2$ and $\zeta_i = \nu \eta_i$, we have

$$\boldsymbol{\beta}_j - \sum_{i=1}^{n} \zeta_i \mathbf{x}_{ij} + \theta \nu \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.26a}$$

$$\sum_{i=1}^{n} \zeta_i = 0, \tag{3.26b}$$

$$y_i - (\sum_{j \in \mathcal{V}} \boldsymbol{\beta}_j \mathbf{x}_{ij} + \beta_0) = 0, \quad i \in \mathcal{E} \tag{3.26c}$$

This linear system consists of $|\mathcal{V}| + |\mathcal{E}| + 1$ equations and $|\mathcal{V}| + |\mathcal{E}| + 1$ unknowns ($\theta = \lambda_1$ is fixed). When ν changes by a small enough amount, the sets $\mathcal{R}$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$ will not change, and the structure of the above system will not change. (3.26) can be written as:

$$\boldsymbol{\beta}_j - \sum_{i \in \mathcal{E}}^{n} \zeta_i \mathbf{x}_{ij} - \sum_{i \notin \mathcal{E}}^{n} \nu \eta_i \mathbf{x}_{ij} + \theta \nu \cdot sign(\boldsymbol{\beta}_j) = 0, \quad j \in \mathcal{V} \tag{3.27a}$$

$$\sum_{i\in\mathcal{E}}^{n}\zeta_i+\sum_{i\notin\mathcal{E}}^{n}\zeta_i=0, \tag{3.27b}$$

$$y_i-(\sum_{j\in\mathcal{V}}\boldsymbol{\beta}_j\mathbf{x}_{ij}+\beta_0)=0, \quad i\in\mathcal{E} \tag{3.27c}$$

Taking derivatives with respect to ν, we have

$$\frac{\Delta\boldsymbol{\beta}_j}{\Delta\nu}-\sum_{i\in\mathcal{E}}\frac{\Delta\zeta_i}{\Delta\nu}\mathbf{x}_{ij}=\sum_{i\notin\mathcal{E}}\eta_i\mathbf{x}_{ij}-\theta\cdot sign(\boldsymbol{\beta}_j), \quad j\in\mathcal{V}$$

$$\sum_{i\in\mathcal{E}}\frac{\Delta\zeta_i}{\Delta\nu}=-\sum_{i\notin\mathcal{E}}\eta_i=\sum_{i\in\mathcal{E}}\eta_i$$

$$\sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{\beta}_j}{\Delta\nu}\mathbf{x}_{ij}+\frac{\Delta\beta_0}{\Delta\nu}=0, \quad i\in\mathcal{E}$$

We can find $\boldsymbol{\beta}$, β_0 and ζ_i are piecewise linear in ν. In (3.26), if we plug in $\boldsymbol{\beta}_j=\frac{1}{\lambda_2}\sum_{i=1}^n\eta_i\mathbf{x}_{ij}-\frac{\theta}{\lambda_2}\cdot sign(\boldsymbol{\beta}_j)$

$$\sum_{i=1}^{n}\eta_i=0,$$

$$y_i\lambda_2-(\sum_{j\in\mathcal{V}}(\sum_{i=1}^{n}\eta_i\mathbf{x}_{ij}-\theta\cdot sign(\boldsymbol{\beta}_j))\mathbf{x}_{ij}+\eta_0)=0, \quad i\in\mathcal{E}$$

where $\eta_0=\lambda_2\beta_0$. Then take derivative with respect to λ_2, we have

$$\sum_{i\in\mathcal{E}}\frac{\Delta\eta_i}{\Delta\lambda_2}=0,$$

$$\sum_{j\in\mathcal{V}}(\sum_{i\in\mathcal{E}}\frac{\Delta\eta_i}{\Delta\lambda_2}\mathbf{x}_{ij})\mathbf{x}_{ij}+\frac{\Delta\eta_0}{\Delta\lambda_2}=y_i, \quad i\in\mathcal{E}$$

This linear system consists of $|\mathcal{E}|+1$ equations and $|\mathcal{E}|+1$ unknowns.We can find that η_i is piecewise linear in λ_2. Then we need to compute the increase of ν (or decrease of λ_2) is needed to reach the next event:

(1) An active feature in $\mathcal{V}$ becomes inactive:

$$\delta_\nu^1=\min_{j\in\mathcal{V}_+}\left\{\frac{0-\boldsymbol{\beta}_j}{\Delta\boldsymbol{\beta}_j/\Delta\nu}\right\}, \tag{3.31}$$

where $\mathcal{V}_+=\{j\in\mathcal{V}:\frac{0-\boldsymbol{\beta}_j}{\Delta\boldsymbol{\beta}_j/\Delta\nu}>0\}$.

(2) An inactive feature joins active set $\mathcal{V}$. Let $f_j^+ = \frac{-\theta - c_j}{\Delta c_j/\Delta\lambda_2}$, $f_j^- = \frac{\theta - c_j}{\Delta c_j/\Delta\lambda_2}$, then we have

$$g_j = \begin{cases} \max\{f_j^+, f_j^-\}, \text{ if } f_j^+ < 0 \text{ and } f_j^- < 0 \\ \min\{f_j^+, f_j^-\}, \text{ if either } f_j^+ \geq 0 \text{ or } f_j^- \geq 0 \text{ but not both} \\ -\infty, \text{ if } f_j^+ \geq 0 \text{ and } f_j^- \geq 0 \end{cases}$$

$$\delta_{\lambda_2}^2 = \max_{j \notin \mathcal{V}} \{g_j\} \tag{3.32}$$

where $\frac{\Delta c_j}{\Delta\lambda_2} = -\sum_{i=1}^n \frac{\Delta\eta_i}{\Delta\lambda_2}\mathbf{x}_{ij}$

(3) A point moves from $\mathcal{E}$ to $\mathcal{L}$ or $\mathcal{R}$:

$$\delta_{\lambda_2}^3 = \max_{i \in \mathcal{E}} \min\left\{\frac{-\frac{1}{n} - \eta_i}{\Delta\eta_i/\Delta\lambda_2}, \frac{\frac{1}{n} - \eta_i}{\Delta\eta_i/\Delta\lambda_2}\right\} \tag{3.33}$$

(4) A point moves from $\mathcal{L}$ or $\mathcal{R}$ to $\mathcal{E}$:

$$\delta_\nu^4 = \min_{i \in \mathcal{E}_+^c} \left\{\frac{0 - r_i}{\Delta r_i/\Delta\nu}\right\}, \tag{3.34}$$

where $\mathcal{E}_+^c = \{i \notin \mathcal{E} : \frac{0 - r_i}{\Delta r_i/\Delta\nu} > 0\}$.

(5) λ_2 reaches to 0:

$$\delta_{\lambda_2}^5 = -\lambda_2 \tag{3.35}$$

Then find the event happens first by

$$\delta_{\lambda_2} = \max\left\{\frac{-\lambda_2^2\delta_\nu^1}{1 + \lambda_2\delta_\nu^1}, \delta_{\lambda_2}^2, \delta_{\lambda_2}^3, \frac{-\lambda_2^2\delta_\nu^4}{1 + \lambda_2\delta_\nu^4}, \delta_{\lambda_2}^5\right\} \tag{3.36}$$

The algorithm for the λ_2-path is shown in Algorithm 3.2.

The solution path algorithm for the DrLAD is shown in Algorithm 3.3.

3.5 Numerical Experiment

3.5.1 Computational efficiency

The experiments were performed on a Linux server with AMD Zen2 16 core processors (2.3GHz) and 64GB RAM. In this experiment, we implemented 5-fold cross-validation

Algorithm 3.2: λ_2-path algorithm

1 Initialize s, λ_2, ν, β_0, $\boldsymbol{\beta}$, η_i, $\mathcal{R}$, $\mathcal{L}$, $\mathcal{E}$, $\mathcal{V}$;
2 Compute $\delta_\nu^1, \delta_{\lambda_2}^2, \delta_{\lambda_2}^3, \delta_\nu^4, \delta_{\lambda_2}^5$ by equations (3.31)-(3.35), and find δ_{λ_2} by equation (3.36);
3 Update $\boldsymbol{\beta}_j, \beta_0, \eta_i$ by
$\boldsymbol{\beta}_j \leftarrow \boldsymbol{\beta}_j + \delta_\nu \frac{\Delta \boldsymbol{\beta}_j}{\Delta \nu}, \quad j \in \mathcal{V}, \qquad \beta_0 \leftarrow \beta_0 + \delta_\nu \frac{\Delta \beta_0}{\Delta \nu}, \quad \eta_i \leftarrow \eta_i + \delta_{\lambda_2} \frac{\Delta \eta_i}{\Delta \lambda_2}, \quad i \in \mathcal{E}$
Update $\mathcal{L}, \mathcal{R}, \mathcal{E}$, and $\mathcal{V}$;
4 Set $\lambda_2 \leftarrow \lambda_2 + \delta_{\lambda_2}$;
5 Stop the algorithm when λ_2 reaches to 0 (or lower bound). Otherwise, return to step 2.

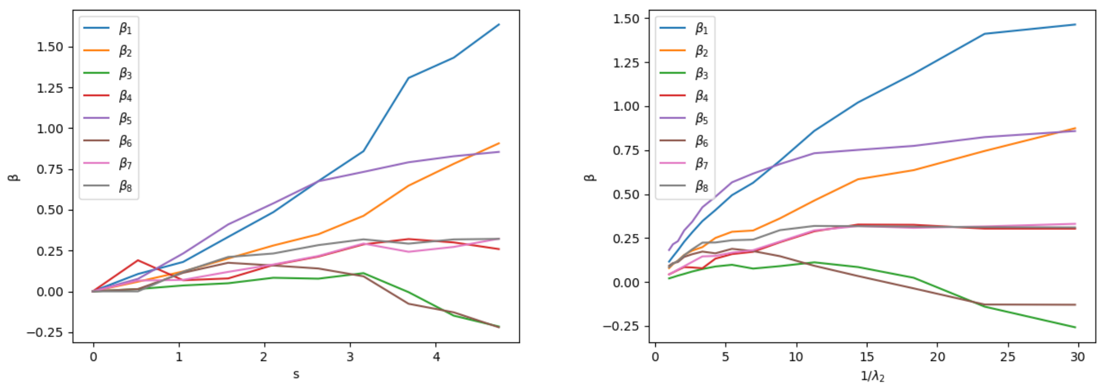
Algorithm 3.3: Solution path algorithm for DrLAD

1 Given a grid values of s and λ_2;
2 Set $\lambda_2 = 0$, Compute the solution path for s using Algorithm 3.1;
3 Set $s = 0$, compute the solution path with respect to λ_2 using Algorithm 3.2;
4 Set $s \leftarrow s + \Delta s$, compute the solution path with respect to λ_2 using Algorithm 3.2;
5 Stop the algorithm when s reaches upper bound. Otherwise, return to step 4.

on Prostate data and Triazines data to select the appropriate hyperparameters. The computational time and test error were recorded for solution path algorithm and CVXPY solver [122]. The experiments were repeated 10 times, and the results are shown in Table 3.1 and Table 3.2.

In Table 3.1, as the number of hyperparameters increase, the computational time will increase. The computational time of CVXPY method scales in total number of hyperparameters. For example, when the number of s is 5, the number of λ_2 is 5, the average computational time is 2.218s. When the number of λ_2 doubles to 10, the computational time also doubles to 4.395s. When the number of s is 10, the number of λ_2 is 5, the computational time is 4.481s. The computational time in fourth row is very close to the time in second row, because they have same total numbers of hyperparameters.

The computational time of solution path algorithm mainly scales in number of



(a) (b)

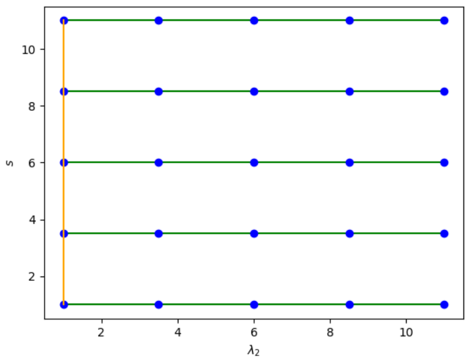
Figure 3.2: The Piecewise Linear Solution Path of Coefficients with Respect to Different Hyperparameters

Table 3.1: Experiments of DrLAD on Prostate Data

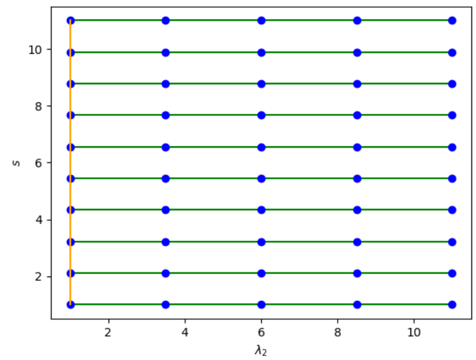
Hyperparameter		Computational time		Test Error	
# of s	# of λ_2	SP method (s)	CVXPY(s)	SP method	CVXPY
5	5	$\mathbf{0.482 \pm 0.078}$	1.709 ± 0.011	0.554 ± 0.204	$\mathbf{0.548 \pm 0.193}$
5	10	$\mathbf{0.497 \pm 0.104}$	3.884 ± 0.041	$\mathbf{0.537 \pm 0.198}$	0.547 ± 0.194
5	20	$\mathbf{0.522 \pm 0.116}$	8.129 ± 0.060	$\mathbf{0.531 \pm 0.195}$	0.550 ± 0.189
10	5	$\mathbf{0.781 \pm 0.185}$	3.462 ± 0.042	$\mathbf{0.547 \pm 0.204}$	0.556 ± 0.204
10	10	$\mathbf{0.873 \pm 0.227}$	7.774 ± 0.065	0.532 ± 0.187	$\mathbf{0.529 \pm 0.170}$
10	20	$\mathbf{0.976 \pm 0.269}$	16.371 ± 0.179	$\mathbf{0.526 \pm 0.188}$	0.538 ± 0.188
20	5	$\mathbf{1.452 \pm 0.316}$	6.905 ± 0.063	$\mathbf{0.538 \pm 0.185}$	0.550 ± 0.200
20	10	$\mathbf{1.666 \pm 0.414}$	15.611 ± 0.144	0.533 ± 0.186	$\mathbf{0.533 \pm 0.185}$
20	20	$\mathbf{1.873 \pm 0.495}$	32.884 ± 0.308	$\mathbf{0.530 \pm 0.185}$	0.545 ± 0.184

s. When the number of λ_2 fixed 5, the number of s increases from 5 to 10, the computational time increases from 0.481s to 0.772s. The time nearly doubles as the number of s doubles. But when the number of s fixed 5, the number of λ_2 increases from 5 to 10, the computational time only increases from 0.481s to 0.525s.

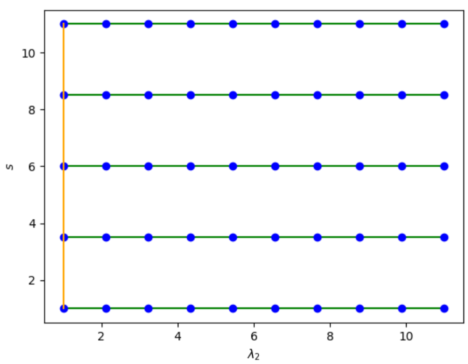
In Figure 3.3, the CVXPY method only computes the optimal solutions with respect to the blue points. So the computational time is scale in the total number of blue points. The solution path algorithm calculates all the optimal solutions along



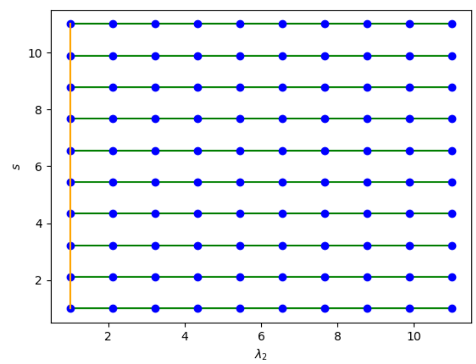
(a) Number of s is 5, Number of λ_2 is 5



(b) Number of s is 10, Number of λ_2 is 5



(c) Number of s is 5, Number of λ_2 is 10



(d) Number of s is 10, Number of λ_2 is 10

Figure 3.3: The Grid of Hyperparameters

the solution path, corresponding the green lines and orange lines. From Figure 3.3a to 3.3b, the number of s increases, we need to calculate the new path for λ_2, which will lead to more computational cost. But from Figure 3.3a to 3.3c, although the number of λ_2 increases, the path remains the same as Figure 3.3a. The increased time is mainly caused by more iterations.

Compared with CVXPY method, solution path algorithm also achieves good test error, which means that it can select appropriate hyperparameters.

Table 3.2 shows the results for Triazines data. It also validates the efficiency of solution path algorithm. The triazines data has higher dimensions (60) than the

Table 3.2: Experiments of DrLAD on Triazines Data

Hyperparameter		Computational time		Test Error	
# of s	# of λ_2	SP method (s)	CVXPY(s)	SP method	CVXPY
5	5	$\mathbf{3.774 \pm 0.226}$	9.266 ± 0.166	$\mathbf{0.0226 \pm 0.0082}$	0.0229 ± 0.0078
5	10	$\mathbf{4.219 \pm 0.208}$	18.067 ± 0.345	$\mathbf{0.0212 \pm 0.0066}$	$\mathbf{0.0212 \pm 0.0066}$
5	20	$\mathbf{4.583 \pm 0.355}$	35.604 ± 0.674	$\mathbf{0.0214 \pm 0.0069}$	$\mathbf{0.0214 \pm 0.0069}$
10	5	$\mathbf{7.147 \pm 0.391}$	18.796 ± 0.343	$\mathbf{0.0200 \pm 0.0061}$	$\mathbf{0.0200 \pm 0.0061}$
10	10	$\mathbf{8.193 \pm 0.532}$	36.446 ± 0.638	$\mathbf{0.0202 \pm 0.0059}$	0.0204 ± 0.0057
10	20	$\mathbf{9.252 \pm 0.622}$	72.497 ± 1.184	$\mathbf{0.0197 \pm 0.0053}$	0.0202 ± 0.0058
20	5	$\mathbf{14.217 \pm 0.911}$	37.670 ± 0.554	$\mathbf{0.0202 \pm 0.0060}$	0.0203 ± 0.0060
20	10	$\mathbf{16.253 \pm 0.954}$	74.473 ± 1.543	$\mathbf{0.0201 \pm 0.0062}$	0.0203 ± 0.0060
20	20	$\mathbf{18.470 \pm 1.444}$	145.494 ± 2.045	$\mathbf{0.0196 \pm 0.0056}$	0.0201 ± 0.0061

prostate data (8). Regularization technique is widely used in high dimensional data analysis. The regularization term can perform feature selection and deal with correlated features. The solution path algorithm highly improves the efficiency of cross-validation process and select the appropriate hyperparameters.

3.5.2 Accuracy

Simulation Experiments.

In this experiment, we generate simulated data with group correlated coefficients under different correlations. The data are generated as follows:

$$\mathbf{x}_i \sim N_m(\mathbf{0}, \mathbf{\Sigma})$$

$$y_i \sim N_m(\mathbf{x}_i^\intercal \boldsymbol{\beta}^*, \sigma^2)$$

where the covariance matrix $\mathbf{\Sigma}$ is defined as

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}^1_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{\Sigma}^2_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{\Sigma}^3_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{21\times21} \end{pmatrix}, \quad \mathbf{\Sigma}^1_{3\times3} = \mathbf{\Sigma}^2_{3\times3} = \mathbf{\Sigma}^3_{3\times3} = \begin{pmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{pmatrix}$$

ρ is the correlation of two variables. The coefficients are defined as

$$\boldsymbol{\beta}^* = [1, 1, 1, 0.5, 0.5, 0.5, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

We use the following evaluation metrics [123]:

- Mean Squared Error (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{x}_i^\intercal \hat{\boldsymbol{\beta}})^2$$

- Relative Risk (RR):

$$RR(\hat{\boldsymbol{\beta}}) = \frac{\mathbb{E}(\mathbf{x}^\intercal \hat{\boldsymbol{\beta}} - \mathbf{x}^\intercal \boldsymbol{\beta}^*)^2}{\mathbb{E}(\mathbf{x}^\intercal \boldsymbol{\beta}^*)^2} = \frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^\intercal \boldsymbol{\Sigma} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)}{(\boldsymbol{\beta}^*)^\intercal \boldsymbol{\Sigma} \boldsymbol{\beta}^*}$$

- Relative Test Error (RTE):

$$RTE(\hat{\boldsymbol{\beta}}) = \frac{\mathbb{E}(\mathbf{y} - \mathbf{x}^\intercal \hat{\boldsymbol{\beta}})^2}{\sigma^2} = \frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^\intercal \boldsymbol{\Sigma} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*) + \sigma^2}{\sigma^2}$$

- Proportion of Variance Explained (PVE):

$$PVE(\hat{\boldsymbol{\beta}}) = 1 - \frac{(\mathbf{y} - \mathbf{x}^\intercal \hat{\boldsymbol{\beta}})^2}{Var(\mathbf{y})} = 1 - \frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^\intercal \boldsymbol{\Sigma} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*) + \sigma^2}{(\boldsymbol{\beta}^*)^\intercal \boldsymbol{\Sigma} \boldsymbol{\beta}^* + \sigma^2}$$

Figure 3.4 shows that DrLAD acheives best performance in RR, RTE and PVE, which means that it provides the best explanation of the estimated coefficients.

In Figure 3.4a, as the correlation increases, the estimation of coefficients become harder, so MSE has an increase trend for all the models. The LAD model has no regularization terms, the generation performance is not good. It has the highest MSE among the four models. The DrLAD model achieves the lowest MSE or second lowest MSE for varied correlations. The elastic net regularization term is a combination of ℓ_1-norm regularization and ℓ_2-norm regularization. So it can perform feature selection like ℓ_1-norm regularization and deal with correlation like ℓ_2-norm regularization. It also encourages grouping effect to select a group of coefficients.

Figure 3.4b is the result of RR. RR is the ratio of expectation of squared error and expectation of true response. The score is 0 if the estimated coefficients equal true coefficients ($\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^*$). It can be found that the RR score of LAD model is very high while DrLAD model has the lowest RR in varied correlations.

RTE in Figure 3.4c is the ratio of expected squared error and Bayes error rate. If $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^*$, the RTE score is 1. DrLAD model obtain the lowest RTE score compared with other models.

PVE is used to measure the proportion to which the model can explain the variation of data. Figure 3.4d shows that the DrLAD achieves the highest PVE scores, which means that the coefficients estimated by DrLAD model account for the data better than the remaining models.

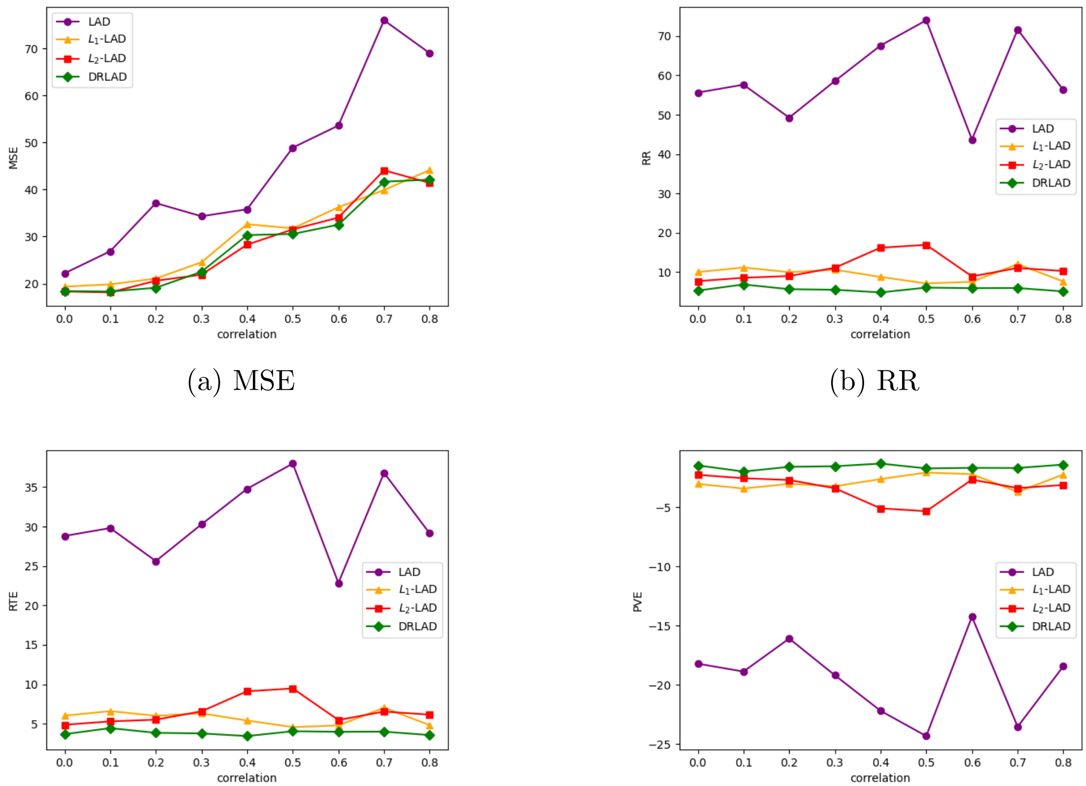
Real data Experiments.

In this experiment, we compare LAD, ℓ_1-LAD, ℓ_2-LAD and DrLAD in different datasets. The information of the datasets are summarized in Table 3.3. Each dataset is split into 80% for training and 20% for test. 5-fold cross-validation is used for parameter tuning.

Table 3.3: Summary of Data Sets

Dataset	Number of samples	Dimensionality
Prostate	97	8
Wine quality	4898	12
Forest fire	517	12
Auto mpg	398	7
Triazines	186	60

In Table 3.4, we can find that the regularized LAD models have better generalization performance compared with LAD model. It validates that the regularization



(a) MSE (b) RR

(c) RTE (d) PVE

Figure 3.4: Results of Simulation Experiments

term can improve the generation performance. ℓ_1-LAD has the lowest MSE in Auto MPG data. ℓ_2-LAD has the best performance in Triazines data. DrLAD has the lowest MSE in Prostate, Wine quality, Forest fire and second lowest MSE in Auto MPG and Triazines. As the combination of ℓ_1 and ℓ_2 regularization, the elastic net regularization term enjoys the benefits from both ℓ_1 and ℓ_2 regularization, and has the better generalization performance in different data sets.

Table 3.4: Mean Squred Error (MSE) of LAD, ℓ_1-LAD, ℓ_2-LAD and DrLAD

Dataset	LAD	ℓ_1-LAD	ℓ_2-LAD	DrLAD
Prostate	0.571 ± 0.210	0.568 ± 0.203	0.539 ± 0.189	$\mathbf{0.526 \pm 0.188}$
Wine quality	0.573 ± 0.025	0.570 ± 0.022	$\mathbf{0.569 \pm 0.022}$	$\mathbf{0.569 \pm 0.022}$
Forest fire	2.563 ± 0.363	2.469 ± 0.427	2.544 ± 0.358	$\mathbf{2.422 \pm 0.457}$
Auto MPG	12.560 ± 1.480	$\mathbf{12.481 \pm 1.460}$	12.548 ± 1.680	12.539 ± 1.680
Triazines	0.0294 ± 0.0084	0.0202 ± 0.0062	$\mathbf{0.0194 \pm 0.0045}$	0.0196 ± 0.0056

3.6 Conclusions

In this chapter, we derived a general form for distributionally robust regression based on distributionally robust optimization theory. This model can generalize many regression models, such as support vector regression, quantile regression. We also derived a new model DrLAD based on this general framework. The optimal solutions of these models are piecewise linear with respect to hyperparameters. A solution path algorithm is proposed to efficiently obtain the optimal solutions and speed up the hyperparameter tuning.

The experiments of computational time show that the solution path algorithm highly improves the cross-validation process compared to directly applying CVXPY solver. In the simulation experiments, we can find that the regularization terms can deal with correlation and the elastic net regularization term has the overall best performance. The real data experiments also show that the regularization terms can decrease the generalization error and the DrLAD model has good performance in different data sets.

For future research, we can extend the solution path algorithm to distributionally robust regression models with more complex loss functions and regularization terms and to distributionally robust classification models.

Chapter 4
Solution Path Algorithm for Double Margin Support Vector Machines

In this chapter, a new SVM model with double regularization terms and double margins is derived based on DRO. The proposed model can explain the data uncertainty in probabilistic way as well as perform automatic feature selection for high dimensional data. We prove that the optimal solutions of this model change piecewise linearly with respect to the hyperparameters. Based on this piecewise linear property, a solution path algorithm is proposed to efficiently obtain the optimal solutions and thus speed up the hyperparameter tuning process. Numerical experiments indicate that the solution path algorithm is very efficient and the proposed model obtain good performance under noisy data.

The remainder of this chapter is organized as follows. Chapter 4.1 introduces the background of this chapter. In Chapter 4.2 presents the derivation of Double Margin Doubly Regularized SVM (DM-DrSVM) and its solution property. In Chapter 4.3, a solution path algorithm is developed for DM-DrSVM. In Chapter 4.4, numerical experiments are performed to test the efficiency of the solution path algorithm and the performance of the DM-DrSVM model. Finally, Chapter 4.5 concludes this chapter.

4.1 Introduction

Support Vector machines (SVM) [60] are popular supervised machine learning methods for classification problems. The main idea of SVM is to find a hyperplane which can maximize the distance between two margins. By introducing slack variables in the constraints, Soft Margin SVM is proposed to deal with the data which is not separable.

In classical SVM, it is assumed that the data points are not corrupted with noise. However, in real applications, the observations are often plagued by uncertainty, such as the measurement error introduced by the limited precision of the instrument, missing data and label noise [124]. In order to estimate the true input-output relationship, noise structure should be considered during model estimation. Robust Optimization is a classical technique to model the data uncertainty. Trafalis et al. [1] and Fan et al. [2] modeled the data uncertainty by bounded uncertainty set. Chance constraint was used by Ben-Tal et al. [3] and Wang et al. [4] to model data uncertainty. Recently, Distributionally Robust Optimization (DRO) is attracting more and more attention in modeling data uncertainty. DRO minimizes the worst-case expected loss, where the expectation is taken with respect to the distributions contained in an ambiguity set. The ambiguity set is defined by moment [6, 125] or probability distance [9, 11].

Many machine learning models are developed based on Wasserstein ambiguity set. Shaeezadeh-Abadeh et al. [15] proposed a distributionally robust logistic regression model. Lee and Mehrotra et al. [16] developed a distributionally robust framework for support vector machines. Shaeezadeh-Abadeh et al. [8] proposed a new regularization technique which can be applied to many regression and classification models based on Wasserstein DRO.

Regarding the solution techniques, cutting surface-based methods were proposed to solve distributionally robust support vector machines by Lee and Mehrotra [16] and Luo and Mehrotra [22]. Blanchet et al. [23] and Sinha et al. [19] proposed stochastic gradient descent algorithms to tackle the Wasserstein DRO problems. For distributionally logistic regression, Li et al. [24] developed an ADMM-based first-order algorithmic framework. Epigraphical projection-based incremental algorithms were proposed by Li et al. [25] to solve distributionally robust support vector machines problem.

However, the above solution techniques are proposed for solving the DRO model with fixed hyperparameters. In Wasserstein DRO models, the radius λ of the Wasserstein ambiguity set is an important hyperparameters. With larger λ, more distributions will be included; with smaller λ, the empirical distribution will have more influence. To select the appropriate λ, we need to solve a series DRO models with different hyperparameters. Directly applying the grid search is computationally intense and very inefficient. Solution path algorithms can be applied to speed up the hyperparameter tuning.

Solution path algorithms can obtain the optimal solutions with respect to different hyperparameters without retraining the model multiple times. These kinds of algorithms have been widely used in classical support vector machines. Hastie et al. [37] firstly introduced the SVMPath algorithm to fit the entire path for SVM solutions with respect to different regularization hyperparameters. Zhu et al. [41] developed an efficient algorithm to compute the solution path of the ℓ_1-SVM. Wang et al. [44] proposed solution path algorithms for doubly regularized SVM. For ν-SVM, efficient algorithms were designed by Gu et al. [42,43] to trace the solution path with respect ν. Washburn et al. [17] developed a solution path algorithm for distributionally robust ν-SVM.

However, there is no efficient algorithms for hyperparameter tuning of distributionally robust support vector machines. In this chapter, we develop a distributionally robust SVM formulations, called doubly regularized double margin support vector machines (DM-DrSVM). Efficient algorithms are proposed to trace the solution paths with respect the two regularization hyperparameters.

4.2 Distributionally Robust Support Vector Machines and Solution Properties

4.2.1 Wasserstein distributionally robust SVM

Let $s = (\boldsymbol{x}, y)$, $\Xi = \mathbb{R}^d \times \mathbb{R}$, the Wasserstein distance between two distributions $\mathbb{Q}$ and $\mathbb{Q}'$ supported on Ξ is defined as

$$W(\mathbb{Q}, \mathbb{Q}') = \inf_{\Xi} \left\{ \int_{\Xi^2} d(s, s')\Pi(ds, ds') : \begin{array}{l} \Pi \text{ is a joint distribution of } s \text{ and } s' \\ \text{with marginals } \mathbb{Q} \text{ and } \mathbb{Q}' \text{ respectively} \end{array} \right\}$$

where $d(s, s') = \|\boldsymbol{x} - \boldsymbol{x}'\| + \frac{\kappa}{2}|y - y'|$. $\kappa > 0$ represents the relative emphasis between feature mismatch and label uncertainty. The empirical distribution for the training data $\{(\hat{\boldsymbol{x}}_i, \hat{y}_i)\}_{i=1}^n$ is $\hat{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^n \delta_{(\hat{\boldsymbol{x}}_i, \hat{y}_i)}$. Then we can construct the Wasserstein ambiguity set around $\hat{\mathbb{P}}$

$$\mathbb{B}_\lambda(\hat{\mathbb{P}}) = \{\mathbb{Q} : W(\mathbb{Q}, \hat{\mathbb{P}}) \le \lambda\}$$

where λ is the radius of the Wasserstein ambiguity set. Then the distributionally robust support vector machine can be formulated as

$$\min_{\boldsymbol{w}, b} \ \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \sup_{\mathbb{Q} \in \mathcal{B}_{\lambda_1}(\hat{\mathbb{P}})} \mathbb{E}^{\mathbb{Q}}[\ell(\boldsymbol{x}, y)], \tag{4.1}$$

where $\ell(\boldsymbol{x}, y) = \max\{0, 1 - y(\boldsymbol{w}^\intercal \boldsymbol{x} + b)\}$ is the hinge loss for SVM. It can be reformulated as

$$\begin{aligned} \min_{\boldsymbol{w}, b, \xi} \ & \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \lambda_1 \rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. \ & 1 - y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) \le \xi_i, \quad i = 1, \dots, n \\ & 1 + y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) - \kappa\rho \le \xi_i, \quad i = 1, \dots, n \\ & \|\boldsymbol{w}\|_1 \le \rho \\ & \xi_i \ge 0, \quad i = 1, \dots, n \\ & \rho \ge 0 \end{aligned}$$

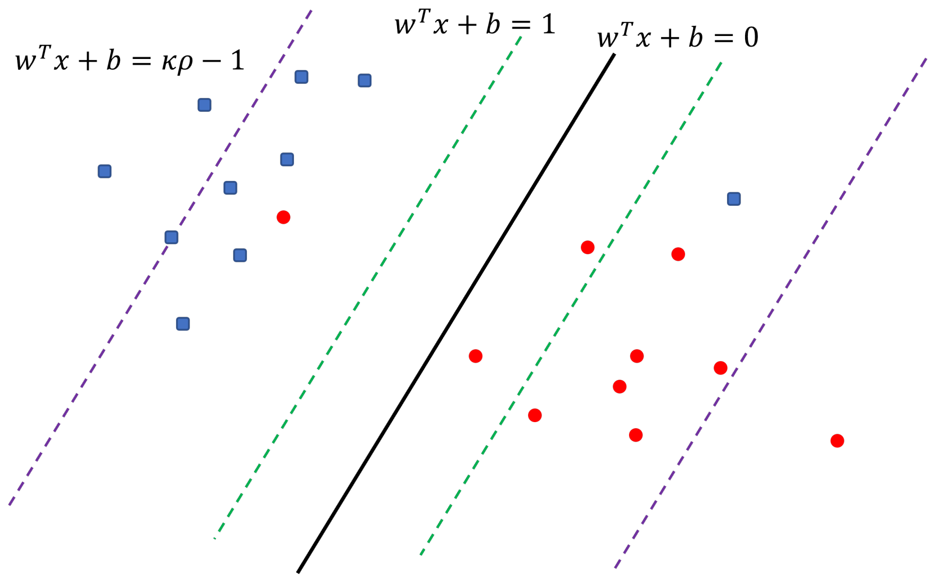


Figure 4.1: Double Margin Doubly Regularized Support Vector Machine (DM-DrSVM)

Since $\rho \geq \|\boldsymbol{w}\|_1 \geq 0$, then we can remove $\rho \geq 0$. Then the above problem can be simplified as

$$\min_{\boldsymbol{w},b,\xi} \ \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \lambda_1 \rho + \frac{1}{n}\sum_{i=1}^{n} \xi_i \tag{4.2a}$$

$$s.t. \ 1 - y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) \leq \xi_i, \quad i = 1, \ldots, n \tag{4.2b}$$

$$1 + y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) - \kappa\rho \leq \xi_i, \quad i = 1, \ldots, n \tag{4.2c}$$

$$\|\boldsymbol{w}\|_1 \leq \rho \tag{4.2d}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, n \tag{4.2e}$$

Different from classical SVM, the derived new SVM has another pair of margin and regularization term introduced by the DRO formulation. Thus, we call this new model as Double Margin Doubly Regularized Support Vector Machine (DM-DrSVM).

4.2.2 Piecewise linear solution path for DM-DrSVM

The model (4.2) has convex quadratic objective function and linear constraints, so it is convex programming problem. Karush–Kuhn–Tucker (KKT) conditions can be applied to this model to obtain the optimal solutions. The Lagrangian function of (4.2) is

$$\begin{aligned} L(\boldsymbol{w}, b, \rho, \theta, \xi, \alpha, \mu, \eta; \lambda_1, \lambda_2) =& \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \lambda_1\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1 - y_i(\boldsymbol{w}^\intercal\boldsymbol{x}_i + b) - \xi_i) \\ &+ \sum_{i=1}^{n}\mu_i(1 + y_i(\boldsymbol{w}^\intercal\boldsymbol{x}_i + b) - \kappa\rho - \xi_i) + \theta(\|\boldsymbol{w}\|_1 - \rho) - \sum_{i=1}^{n}\eta_i\xi_i \end{aligned}$$

The KKT conditions are

$$\begin{aligned} &\frac{\partial L}{\partial \boldsymbol{w}_j} = \lambda_2\boldsymbol{w}_j + \sum_{i=1}^{n} -\alpha_i y_i \boldsymbol{x}_{ij} + \mu_i y_i \boldsymbol{x}_{ij} + \theta \cdot sign(\boldsymbol{w}_j) = 0, \quad j \in \mathcal{V} \\ &\frac{\partial L}{\partial b} = \sum_{i=1}^{n} -\alpha_i y_i + \mu_i y_i = 0 \\ &\frac{\partial L}{\partial \xi_i} = \frac{1}{n} - \alpha_i - \mu_i - \eta_i = 0 \\ &\frac{\partial L}{\partial \rho} = \lambda_1 - \theta - \sum_{i=1}^{n}\mu_i\kappa = 0 \\ &\alpha_i(1 - y_i(\boldsymbol{w}^\intercal\boldsymbol{x}_i + b) - \xi_i) = 0 \\ &\mu_i(1 + y_i(\boldsymbol{w}^\intercal\boldsymbol{x}_i + b) - \rho\kappa - \xi_i) = 0 \\ &\theta(\|\boldsymbol{w}\|_1 - \rho) = 0 \\ &\eta_i\xi_i = 0, \quad i = 1, \ldots, n \end{aligned}$$

Let $f(\boldsymbol{x}_i) = \boldsymbol{w}^\intercal\boldsymbol{x}_i + b$, we define the following sets

$$\begin{aligned} &\mathcal{L} = \{i : y_i f(\boldsymbol{x}_i) < 1\}, \quad \xi_i = 1 - y_i f(\boldsymbol{x}_i) > 0, \alpha_i = \frac{1}{n}, \mu_i = \eta_i = 0 \\ &\mathcal{E}_L = \{i : y_i f(\boldsymbol{x}_i) = 1\}, \quad \xi_i = 0, \alpha_i + \eta_i = \frac{1}{n}, \mu_i = 0 \\ &\mathcal{E}_M = \{i : 1 < y_i f(\boldsymbol{x}_i) < \kappa\rho - 1\}, \quad \xi_i = 0, \eta_i = \frac{1}{n}, \alpha_i = \eta_i = \mu_i = 0 \end{aligned}$$

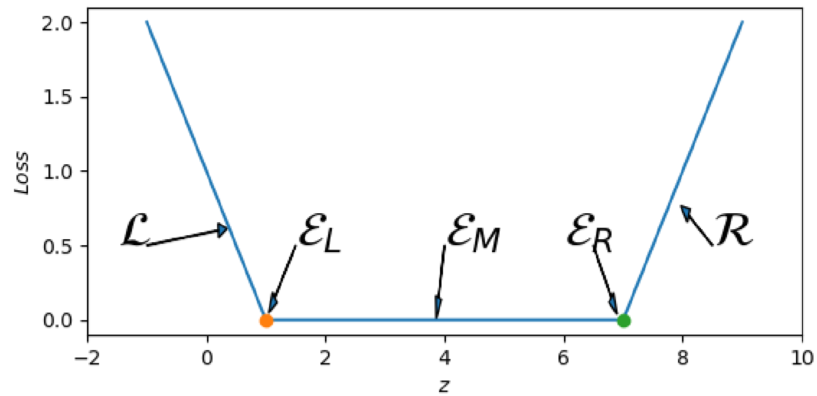


Figure 4.2: Split the Data Points into Different Sets

$$\mathcal{E}_R = \{i : y_i f(\boldsymbol{x}_i) = \kappa\rho - 1\}, \quad \xi_i = 0, \eta_i + \mu_i = \frac{1}{n}, \alpha_i = 0$$
$$\mathcal{R} = \{i : y_i f(\boldsymbol{x}_i) > \kappa\rho - 1\}, \quad \xi_i = 1 + y_i f(\boldsymbol{x}_i) - \kappa\rho > 0, \mu_i = \frac{1}{n}, \alpha_i = \eta_i = 0$$

Then we have the following equations

$$\lambda_2 \boldsymbol{w}_j + \sum_{i=1}^{n} (\mu_i - \alpha_i) y_i \boldsymbol{x}_{ij} + \theta \cdot sign(\boldsymbol{w}_j) = 0, \quad j \in \mathcal{V}$$
$$\sum_{i=1}^{n} (\mu_i - \alpha_i) y_i = 0$$
$$\theta + \sum_{i=1}^{n} \mu_i \kappa = \lambda_1$$
$$\sum_{j \in \mathcal{V}} \boldsymbol{w}_j \cdot sign(\boldsymbol{w}_j) - \rho = 0, \text{ if } \theta > 0$$
$$y_i (\sum_{j \in \mathcal{V}} \boldsymbol{w}_j \boldsymbol{x}_{ij} + b) = 1, \quad i \in \mathcal{E}_L$$
$$1 + y_i (\sum_{j \in \mathcal{V}} \boldsymbol{w}_j \boldsymbol{x}_{ij} + b) - \kappa\rho = 0, \quad i \in \mathcal{E}_R$$

In the above system of equations, we have $|\mathcal{V}| + |\mathcal{E}_L| + |\mathcal{E}_R| + 3$ equations, and $|\mathcal{V}| + |\mathcal{E}_L| + |\mathcal{E}_R| + 3$ unknown variables(θ, γ, b, $\boldsymbol{w}_j$ for $j \in \mathcal{V}$, α_i for $i \in \mathcal{E}_L$, μ_m for $m \in \mathcal{E}_R$). Taking derivative with respect to λ_1, then we have

$$\lambda_2 \frac{\Delta \boldsymbol{w}_j}{\Delta \lambda_1} - \sum_{i \in \mathcal{E}_L} \frac{\Delta \alpha_i}{\Delta \lambda_1} y_i \boldsymbol{x}_{ij} + \sum_{i \in \mathcal{E}_R} \frac{\Delta \mu_i}{\Delta \lambda_1} y_i \boldsymbol{x}_{ij} + \frac{\Delta \theta}{\Delta \lambda_1} \cdot sign(\boldsymbol{w}_j) = 0, \quad j \in \mathcal{V} \tag{4.3a}$$

$$\sum_{i\in\mathcal{E}_R}\frac{\Delta\mu_i}{\Delta\lambda_1}y_i-\sum_{i\in\mathcal{E}_L}\frac{\Delta\alpha_i}{\Delta\lambda_1}y_i=0 \tag{4.3b}$$

$$\frac{\Delta\theta}{\Delta\lambda_1}+\sum_{i\in\mathcal{E}_R}\frac{\Delta\mu_i}{\Delta\lambda_1}\kappa=1 \tag{4.3c}$$

$$\sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{w}_j}{\Delta\lambda_1}\cdot sign(\boldsymbol{w}_j)-\frac{\Delta\rho}{\Delta\lambda_1}=0,\ \text{if } \theta>0 \tag{4.3d}$$

$$y_i(\sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{w}_j}{\Delta\lambda_1}\boldsymbol{x}_{ij}+\frac{\Delta b}{\Delta\lambda_1})=0,\quad i\in\mathcal{E}_L \tag{4.3e}$$

$$y_i(\sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{w}_j}{\Delta\lambda_1}\boldsymbol{x}_{ij}+\frac{\Delta b}{\Delta\lambda_1})-\frac{\Delta\rho}{\Delta\lambda_1}\kappa=0,\quad i\in\mathcal{E}_R \tag{4.3f}$$

The solutions to the system of equations will not depend on λ_1. If $\mathcal{V}$, $\mathcal{E}_L$, $\mathcal{E}_R$ do not change, when λ_1 changes, the variables(θ,γ,b, $\boldsymbol{w}_j$ for $j\in\mathcal{V}$, α_i for $i\in\mathcal{E}_L$, μ_m for $m\in\mathcal{E}_R$) will change linearly with respect to λ_1. Following similar idea, we can also prove that the optimal solutions will change linearly with respect to λ_2.

4.3 Solution Path Algorithm for DM-DrSVM

Since the optimal solutions of DM-DrSVM is piecewise linear in the hyperparameters (λ_1,λ_2), we can develop solution path algorithm to efficiently obtain the optimal solutions.

4.3.1 Solution path algorithm with respect to λ_2

We first set $\lambda_1=0$, and let ρ be a large positive number, the optimization problem becomes:

$$\min_{\boldsymbol{w},b,\xi}\ \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2+\frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{4.4a}$$

$$s.t.\ 1-y_i(\boldsymbol{w}^\intercal\boldsymbol{x}_i+b)\le\xi_i,\quad i=1,\dots,n \tag{4.4b}$$

$$\xi_i\ge 0,\quad i=1,\dots,n \tag{4.4c}$$

When $\lambda_2 = +\infty$, $\boldsymbol{w} = 0$, the above problem becomes

$$\min_{b,\xi} \ \frac{1}{n}\sum_{i=1}^{n} \xi_i \tag{4.5a}$$

$$s.t.\ 1 - y_i b \leq \xi_i, \quad i = 1, \ldots, n \tag{4.5b}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, n \tag{4.5c}$$

Define $\mathcal{I}_+ = \{i : y_i = 1\}$ and $\mathcal{I}_- = \{i : y_i = -1\}$, then we have $n_+ = |\mathcal{I}_+|, n_- = |\mathcal{I}_-|$.

If $n_+ = n_-$, as λ_2 decreases from $+\infty$, one or more positive and negative points will join $\mathcal{E}_L$ simultaneously. The Lagrange function of model (4.4) is

$$L(\boldsymbol{w}, b, \xi, \alpha; \lambda_2) = \frac{\lambda_2}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1 - y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) - \xi_i) + \sum_{i=1}^{n}\eta_i\xi_i$$

The KKT conditions are

$$\begin{aligned}
&\frac{\partial L}{\partial \boldsymbol{w}_j} = \lambda_2 \boldsymbol{w}_j + \sum_{i=1}^{n} -\alpha_i y_i \boldsymbol{x}_{ij} = 0, \quad j \in \mathcal{V} \\
&\frac{\partial L}{\partial b} = \sum_{i=1}^{n} -\alpha_i y_i = 0 \\
&\frac{\partial L}{\partial \xi_i} = \frac{1}{n} - \alpha_i - \eta_i = 0 \\
&\alpha_i(1 - y_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i + b) - \xi_i) = 0, \quad i = 1, \ldots, n \\
&\eta_i \xi_i = 0, \quad i = 1, \ldots, n
\end{aligned}$$

Let i_+ and i_- be the index of the two points join $\mathcal{E}_L$, $f(\boldsymbol{x}) = \boldsymbol{w}^\intercal \boldsymbol{x} + b$, then i_+ and i_- can be obtained by $i_+ = \max_{i \in \mathcal{I}_+} f(\boldsymbol{x}_i)$ and $i_- = \max_{i \in \mathcal{I}_-} f(\boldsymbol{x}_i)$. Since $i_+, i_- \in \mathcal{E}_L$, we can obtain

$$\begin{aligned}
&\begin{cases} \boldsymbol{w}^\intercal \boldsymbol{x}_{i_+} + b = 1 \\ \boldsymbol{w}^\intercal \boldsymbol{x}_{i_-} + b = -1 \end{cases} \Rightarrow \begin{cases} \frac{1}{\lambda_2}(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_+} + b = 1 \\ \frac{1}{\lambda_2}(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_-} + b = -1 \end{cases} \\
&\Rightarrow \begin{cases} \lambda_2 = \frac{(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_+} - (\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i)^\intercal \boldsymbol{x}_{i_-}}{2} \\ b = -\frac{(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_+} + (\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_-}}{(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_+} - (\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i)^\intercal \boldsymbol{x}_{i_-}} \end{cases}
\end{aligned}$$

So we obtain the initial value of λ_2 and b.

If $n_+ > n_-$, the dual formulation of model (4.4) is

$$\begin{aligned}\max_{\alpha} \ & \sum_{i=1}^{n} \alpha_i - \frac{1}{2\lambda_2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j \\ s.t. \ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq \frac{1}{n}\end{aligned}$$

If we set $b = 1$, then we have $\alpha_i = 1/n, i \in \mathcal{I}_-$, and $\sum_{i=1}^{n} \alpha_i = 2n_-/n$. So the above optimization problem can be written as

$$\begin{aligned}\min_{\alpha_i} \ & \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j \\ s.t. \ & 0 \leq \alpha_i \leq \frac{1}{n}, i \in \mathcal{I}_+ \\ & \sum_{i \in \mathcal{I}_+}^{n} \alpha_i = \frac{n_-}{n} \\ & \alpha_i = \frac{1}{n}, i \in \mathcal{I}_-\end{aligned}$$

By solving the above optimization problem, we can obtain the initial values for α_i.

Then we need to find the event when active sets change. Based on the KKT conditions, we have the following equations

$$\begin{aligned}\lambda_2 \boldsymbol{w}_j + \sum_{i=1}^{n} -\alpha_i y_i \boldsymbol{x}_{ij} &= 0, \quad j \in \mathcal{V} \\ \sum_{i=1}^{n} \alpha_i y_i &= 0, \\ y_i(\sum_{j \in \mathcal{V}} \boldsymbol{w}_j \boldsymbol{x}_{ij} + b) &= 1, \quad i \in \mathcal{E}_L\end{aligned}$$

Let $\nu = 1/\lambda_2$, $\alpha_i^* = \alpha_i/\lambda_2$, then we have

$$\boldsymbol{w}_j - \sum_{i=1}^{n} \alpha_i^* y_i \boldsymbol{x}_{ij} = 0, \quad j \in \mathcal{V}$$

$$\sum_{i\in\mathcal{E}_L}\alpha_i^* y_i + \sum_{i\notin\mathcal{E}_L}\alpha_i^* y_i = 0,$$
$$y_i(\sum_{j\in\mathcal{V}}\boldsymbol{w}_j\boldsymbol{x}_{ij} + b) = 1, \quad i\in\mathcal{E}_L$$

Taking derivative with respect to ν, then we have

$$\frac{\Delta\boldsymbol{w}_j}{\Delta\nu} - \sum_{i\in\mathcal{E}_L}\frac{\Delta\alpha_i^*}{\Delta\nu}y_i\boldsymbol{x}_{ij} = \sum_{i\notin\mathcal{E}_L}^{n}\alpha_i y_i\boldsymbol{x}_{ij}, \quad j\in\mathcal{V}$$
$$\sum_{i\in\mathcal{E}_L}\frac{\Delta\alpha_i^*}{\Delta\nu}y_i = \sum_{i\in\mathcal{E}_L}\alpha_i y_i$$
$$\sum_{j\in\mathcal{V}}\frac{\Delta\boldsymbol{w}_j}{\Delta\nu}\boldsymbol{x}_{ij} + \frac{\Delta b}{\Delta\nu} = 0, \quad i\in\mathcal{E}_L$$

The above system of equations have $|\mathcal{V}|+|\mathcal{E}_L|+1$ equations and $|\mathcal{V}|+|\mathcal{E}_L|+1$ unknown variables. From the above equations, we have

$$\boldsymbol{w}_j = \frac{1}{\lambda_2}\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{x}_{ij}$$

The we can obtain

$$\sum_{i=1}^{n}\alpha_i y_i = 0,$$
$$y_i(\sum_{j\in\mathcal{V}}(\frac{1}{\lambda_2}\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{x}_{ij})\boldsymbol{x}_{ij} + b) = 1, \quad i\in\mathcal{E}_L$$

Let $\alpha_0 = b\lambda_2$, we can obtain

$$\sum_{i=1}^{n}\alpha_i y_i = 0,$$
$$y_i(\sum_{j\in\mathcal{V}}(\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{x}_{ij})\boldsymbol{x}_{ij} + \alpha_0) = \lambda_2, \quad i\in\mathcal{E}_L$$

Taking derivative with respect to λ_2, then we have

$$\sum_{i\in\mathcal{E}} \frac{\Delta\alpha_i}{\Delta\lambda_2} y_i = 0,$$

$$\sum_{j\in\mathcal{V}} (\sum_{i\in\mathcal{E}_L} \frac{\Delta\alpha_i}{\Delta\lambda_2} y_i \boldsymbol{x}_{ij}) \boldsymbol{x}_{ij} + \frac{\Delta\alpha_0}{\Delta\lambda_2} = \frac{1}{y_i}, \quad i \in \mathcal{E}_L$$

The above system of equations have $|\mathcal{E}_L|+1$ equations and $|\mathcal{E}_L|+1$ unknown variables. Define $r_i = 1 - y_i f(\boldsymbol{x}_i)$, then we have

$$\frac{\Delta r_i}{\Delta\nu} = -y_i(\sum_{j\in\mathcal{V}} \frac{\Delta \boldsymbol{w}_j}{\Delta\nu} \boldsymbol{x}_{ij} + \frac{\Delta b}{\Delta\nu})$$

Then we decrease the value of λ_2 and one of the following events will happen.

- a point moves from $\mathcal{E}_L$ to $\mathcal{L}$ or $\mathcal{E}_M$

$$\delta^1_{\lambda_2} = \max_{i\in\mathcal{E}_L} \min \left\{ \frac{0-\alpha_i}{\Delta\alpha_i/\Delta\lambda_2}, \frac{\frac{1}{n}-\alpha_i}{\Delta\alpha_i/\Delta\lambda_2} \right\} \tag{4.7}$$

- a point moves from $\mathcal{L}$ or $\mathcal{E}_M$ to $\mathcal{E}_L$

$$\delta^2_{\nu} = \min_{i\in\mathcal{E}^c_+} \left\{ \frac{0-r_i}{\Delta r_i/\Delta\nu} \right\}, \tag{4.8}$$

- λ_2 reaches to 0:

$$\delta^3_{\lambda_2} = -\lambda_2 \tag{4.9}$$

The solution path algorithm with respect to λ_2 is shown in Algorithm 4.1.

4.3.2 Solution path algorithm with respect to λ_1

For given λ_2, we find the entire solution path with respect to λ_1. Denote $r^{in}_i = 1-y_i f(\boldsymbol{x}_i)$ be the residual for inner margin and $r^{out}_i = 1+y_i f(\boldsymbol{x}_i)-\kappa\rho$ be the residual

Algorithm 4.1: Solution path algorithm with respect to λ_2

Initialize λ_2, ν, b, $\boldsymbol{w}$, α_i, $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{V}$
while $\lambda_2 \geq 0$ **do**
    Compute $\delta_\nu^1, \delta_{\lambda_2}^2, \delta_{\lambda_2}^3, \delta_\nu^4, \delta_{\lambda_2}^5$ by equations (4.7)-(4.9)
    Find $\delta_{\lambda_2} = \max(\delta_{\lambda_2}^1, -\frac{\lambda_2^2\delta_\nu^2}{1+\lambda_2\delta_\nu^2}, \delta_{\lambda_2}^3)$;
    Update $\boldsymbol{w}_j, b, \alpha_i$ by

$$\boldsymbol{w}_j \leftarrow \boldsymbol{w}_j + \delta_\nu \frac{\Delta \mathbf{w}_j}{\Delta \nu}, \quad j \in \mathcal{V}, \qquad b \leftarrow b + \delta_\nu \frac{\Delta b}{\Delta \nu}, \qquad \alpha_i \leftarrow \alpha_i + \delta_{\lambda_2} \frac{\Delta \alpha_i}{\Delta \lambda_2}, \quad i \in \mathcal{E}_L$$

    Update $\mathcal{L}$, $\mathcal{E}_L$ and $\mathcal{E}_M$ $\lambda_2 \leftarrow \lambda_2 + \delta_{\lambda_2}$

for outer margin. λ_1 is initialized to be 0, as λ_1 increases, one of the following events will happen, to facility the calculation of δ_{λ_1}, we define a function $g(\delta^-, \delta^+)$

$$g(\delta^-, \delta^+) = \begin{cases} \min\{\delta^-, \delta^+\}, \text{ if } \delta^- \geq 0 \text{ and } \delta^+ \geq 0 \\ +\infty, \text{ if } \delta^- < 0 \text{ and } \delta^+ < 0 \\ \max\{\delta^-, \delta^+\}, \text{ otherwise} \end{cases}$$

(1) a point moves from $\mathcal{E}_L$ to $\mathcal{L}$ or $\mathcal{E}_M$,

$$\delta_{\lambda_1}^1 = \min_{i \in \mathcal{E}_L} g_i(\delta_i^-, \delta_i^+) \tag{4.10}$$

where $\delta_i^- = \frac{0-\alpha_i}{\Delta\alpha_i/\Delta\lambda_1}$, $\delta_i^+ = \frac{1/n-\alpha_i}{\Delta\alpha_i/\Delta\lambda_1}$

(2) a point moves from $\mathcal{L}$ or $\mathcal{E}_M$ to $\mathcal{E}_L$

$$\delta_{\lambda_1}^2 = \min_{i \in (\mathcal{L} \cup \mathcal{E}_M)_+} \left\{ \frac{0 - r_i^{in}}{\Delta r_i^{in}/\Delta\lambda_1} \right\}, \tag{4.11}$$

where $\frac{\Delta r_i^{in}}{\Delta\lambda_1} = -y_i(\sum_{j\in\mathcal{V}} \frac{\Delta \boldsymbol{w}_j}{\Delta\lambda_1}\boldsymbol{x}_{ij} + \frac{\Delta b}{\Delta\lambda_1})$, $(\mathcal{L} \cup \mathcal{E}_M)_+ = \{i \in (\mathcal{L} \cup \mathcal{E}_M) : \frac{0-r_i^{in}}{\Delta r_i^{in}/\Delta\lambda_1} > 0\}$

(3) a point moves from $\mathcal{E}_R$ to $\mathcal{R}$ or $\mathcal{E}_M$

$$\delta_{\lambda_1}^3 = \min_{i \in \mathcal{E}_R} g_i(\delta_i^-, \delta_i^+), \tag{4.12}$$

where $\delta_i^- = \frac{0-\mu_i}{\Delta\mu_i/\Delta\lambda_1}$, $\delta_i^+ = \frac{1/n-\mu_i}{\Delta\mu_i/\Delta\lambda_1}$

(4) a point moves from $\mathcal{R}$ or $\mathcal{E}_M$ to $\mathcal{E}_R$

$$\delta^4_{\lambda_1} = \min_{i \in \mathcal{E}^c_+} \left\{ \frac{0 - r^{out}_i}{\Delta r^{out}_i / \Delta\lambda_1} \right\}, \tag{4.13}$$

where $\frac{\Delta r^{out}_i}{\Delta\lambda_1} = y_i(\sum_{j\in\mathcal{V}} \frac{\Delta \boldsymbol{w}_j}{\Delta\lambda_1}\boldsymbol{x}_{ij} + \frac{\Delta b}{\Delta\lambda_1}) - \kappa\frac{\Delta\rho}{\Delta\lambda_1}$

(5) An active feature in $\mathcal{V}$ becomes inactive:

$$\delta^5_{\lambda_1} = \min_{j \in \mathcal{V}_+} \left\{ \frac{0 - \boldsymbol{w}_j}{\Delta \boldsymbol{w}_j / \Delta\lambda_1} \right\}, \tag{4.14}$$

where $\mathcal{V}_+ = \{j \in \mathcal{V} : \frac{0-\boldsymbol{w}_j}{\Delta\boldsymbol{w}_j/\Delta s} > 0\}$.

(6) An inactive feature joins active set $\mathcal{V}$. This happens when the absolute value of the generalized correlation $c_j = \lambda_2\boldsymbol{w}_j + \sum_{i=1}^n(\mu_i - \alpha_i)y_i\boldsymbol{x}_{ij}$ reaches θ.

$$\delta^6_{\lambda_1} = \min_{j \notin \mathcal{V}} g_j(\delta^-_j, \delta^+_j) \tag{4.15}$$

where $\frac{\Delta c_j}{\Delta\lambda_1} = \sum_{i\in\mathcal{E}_R}\frac{\Delta\mu_i}{\Delta\lambda_1}y_i\boldsymbol{x}_{ij} - \sum_{i\in\mathcal{E}_L}\frac{\Delta\alpha_i}{\Delta\lambda_1}y_i\boldsymbol{x}_{ij}$, $\delta^-_j = \frac{-\theta-c_j}{\Delta c_j/\Delta\lambda_1+\Delta\theta/\Delta\lambda_1}$, $\delta^+_j = \frac{\theta-c_j}{\Delta c_j/\Delta s-\Delta\theta/\Delta\lambda_1}$

(7) λ_1 reaches to upper bound:

$$\delta^7_{\lambda_1} = \lambda^{max}_1 - \lambda_1 \tag{4.16}$$

The solution path algorithm with respect to λ_1 is shown in Algorithm 4.2.

Then we can obtain the solution path algorithm for DM-DrSVM shown in Algorithm 4.3.

4.3.3 Computational complexity

The main computational burden in the λ_1 path is solving the system of equations (4.3). If plug (4.3a) into (4.3d), (4.3e), (4.3f), the size of this linear system is $|\mathcal{E}_L| +$

Algorithm 4.2: Solution path algorithm with respect to λ_1

Initialize λ_1, $\boldsymbol{w}$, b, ρ, θ, α, μ, $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$, $\mathcal{V}$

while $\lambda_1 \leq \lambda_{max}$ **do**

Compute $\delta^1_{\lambda_1}, \delta^2_{\lambda_1}, \delta^3_{\lambda_1}, \delta^4_{\lambda_1}, \delta^5_{\lambda_1}, \delta^6_{\lambda_1}, \delta^7_{\lambda_1}$ by equations (4.10)-(4.16)

Find $\delta_{\lambda_1} = \min(\delta^1_{\lambda_1}, \delta^2_{\lambda_1}, \delta^3_{\lambda_1}, \delta^4_{\lambda_1}, \delta^5_{\lambda_1}, \delta^6_{\lambda_1}, \delta^7_{\lambda_1})$;

Update $\boldsymbol{w}_j, b, \rho, \theta, \alpha_i, \mu_i$ by

$$\begin{aligned}
&\boldsymbol{w}_j \leftarrow \boldsymbol{w}_j + \delta_{\lambda_1}\frac{\Delta \boldsymbol{w}_j}{\Delta\lambda_1}, \quad j \in \mathcal{V}, \qquad b \leftarrow b + \delta_{\lambda_1}\frac{\Delta b}{\Delta\lambda_1}, \qquad ,\\
&\theta \leftarrow \theta + \delta_{\lambda_1}\frac{\Delta\theta}{\Delta\lambda_1}, \qquad \alpha_i \leftarrow \alpha_i + \delta_{\lambda_2}\frac{\Delta\alpha_i}{\Delta\lambda_1}, \quad i \in \mathcal{E}_L,\\
&\mu_i \leftarrow \mu_i + \delta_{\lambda_1}\frac{\Delta\mu_i}{\Delta\lambda_1}, \quad i \in \mathcal{E}_R, \qquad \rho \leftarrow \rho + \delta_{\lambda_1}\frac{\Delta\rho}{\Delta\lambda_1}
\end{aligned}$$

Update $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$, $\mathcal{V}$ $\lambda_1 \leftarrow \lambda_1 + \delta_{\lambda_1}$

$|\mathcal{E}_R| + 3$. So the computational cost is $O((|\mathcal{E}_L| + |\mathcal{E}_R|)^3)$. Since for two consecutive steps, the $\mathcal{E}_L$ and $\mathcal{E}_R$ only differ one observation, the computational cost can be reduced to $O((|\mathcal{E}_L| + |\mathcal{E}_R|)^2)$ via inverse updating/downdating. The computational cost of $\frac{\Delta \boldsymbol{w}_j}{\Delta\lambda_1}$ in (4.3a) is $O((|\mathcal{E}_L|+|\mathcal{E}_R|)\cdot|\mathcal{V}|)$. Assume $|\mathcal{E}_L|+|\mathcal{E}_R| = m$, since $|\mathcal{V}| \leq d$, the computational cost at each step is $O(m^2+dm)$. The number of steps is typically some small multiple c of n. Then the computational burden of λ_1 path is $O(cnm^2+cndm)$.

Similarly, for the λ_2 path, the computational burden is $O(cnm^2 + cndm)$. So the computational cost of the two dimensional solution algorithm is $O((cnm^2 + cndm)^2)$.

4.4 Numerical Experiment

The experiments were performed on a Linux server with AMD Zen2 10 core processors (2.3GHz) and 50GB RAM. We use Diabetes and SCADI data from UCI machine learning repository to test the computational efficiency for solution path algorithms. 5-fold cross-validation is implemented to select the appropriate hyperparameters for DM-DrSVM. The results are shown in Table 4.1 and Table 4.2.

Algorithm 4.3: Solution path algorithm for DM-DrSVM

Initialize λ_1, λ_2, $\boldsymbol{w}$, b, ρ, θ, α, μ, $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$, $\mathcal{V}$

while $\lambda_2 \geq 0$ **do**

  Compute $\delta_\nu^1, \delta_{\lambda_2}^2, \delta_{\lambda_2}^3, \delta_\nu^4, \delta_{\lambda_2}^5$ by equations (4.7)-(4.9)

  Find $\delta_{\lambda_2} = \max(\delta_{\lambda_2}^1, -\frac{\lambda_2^2\delta_\nu^2}{1+\lambda_2\delta_\nu^2}, \delta_{\lambda_2}^3)$;

  Update $\boldsymbol{w}_j, b, \alpha_i$ by

$$\begin{aligned} &\boldsymbol{w}_j \leftarrow \boldsymbol{w}_j + \delta_\nu \frac{\Delta \mathbf{w}_j}{\Delta \nu}, \quad j \in \mathcal{V}, \qquad b \leftarrow b + \delta_\nu \frac{\Delta b}{\Delta \nu}, \\ &\alpha_i \leftarrow \alpha_i + \delta_{\lambda_2} \frac{\Delta \alpha_i}{\Delta \lambda_2}, \quad i \in \mathcal{E}_L \end{aligned}$$

  Update $\mathcal{L}$, $\mathcal{E}_L$ and $\mathcal{E}_M$

  while $\lambda_1 \leq \lambda_{max}$ **do**

    Compute $\delta_{\lambda_1}^1, \delta_{\lambda_1}^2, \delta_{\lambda_1}^3, \delta_{\lambda_1}^4, \delta_{\lambda_1}^5, \delta_{\lambda_1}^6, \delta_{\lambda_1}^7$ by equations (4.10)-(4.16)

    Find $\delta_{\lambda_1} = \min(\delta_{\lambda_1}^1, \delta_{\lambda_1}^2, \delta_{\lambda_1}^3, \delta_{\lambda_1}^4, \delta_{\lambda_1}^5, \delta_{\lambda_1}^6, \delta_{\lambda_1}^7)$;

    Update $\boldsymbol{w}_j, b, \rho, \theta, \alpha_i, \mu_i$ by

$$\begin{aligned} &\boldsymbol{w}_j \leftarrow \boldsymbol{w}_j + \delta_{\lambda_1} \frac{\Delta \boldsymbol{w}_j}{\Delta \lambda_1}, \quad j \in \mathcal{V}, \qquad b \leftarrow b + \delta_{\lambda_1} \frac{\Delta b}{\Delta \lambda_1}, \qquad , \\ &\theta \leftarrow \theta + \delta_{\lambda_1} \frac{\Delta \theta}{\Delta \lambda_1}, \qquad \alpha_i \leftarrow \alpha_i + \delta_{\lambda_2} \frac{\Delta \alpha_i}{\Delta \lambda_1}, \quad i \in \mathcal{E}_L, \\ &\mu_i \leftarrow \mu_i + \delta_{\lambda_1} \frac{\Delta \mu_i}{\Delta \lambda_1}, \quad i \in \mathcal{E}_R, \qquad \rho \leftarrow \rho + \delta_{\lambda_1} \frac{\Delta \rho}{\Delta \lambda_1} \end{aligned}$$

    Update $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$, $\mathcal{V}$

    $\lambda_1 \leftarrow \lambda_1 + \delta_{\lambda_1}$

  $\lambda_2 \leftarrow \lambda_2 + \delta_{\lambda_2}$

Table 4.1 shows the results for diabetes data. The dimension of this data is 8, and the number of data points is 768. It can be found that the solution path algorithm is much faster than directly applying the CVXPY solver. As the number of hyperparameters increases, the time of CVXPY solver will scale in both the number of λ_1 and the number of λ_2. This is because the time used by the CVXPY solver is determined by the number of optimization problems. However, the time of the SP method only scale in the number of λ_2. For given λ_2, we have obtained the entire

Table 4.1: Experiments of DM-DrSVM on Diabetes Data

Hyperparameter		Computational time		Test AUC Score	
# of λ_2	# of λ_1	SP method (s)	CVXPY(s)	SP method	CVXPY
5	5	$\mathbf{13.938 \pm 1.603}$	170.659 ± 13.452	$\mathbf{73.050 \pm 2.845}$	73.317 ± 3.164
5	10	$\mathbf{14.081 \pm 2.475}$	335.690 ± 22.554	$\mathbf{73.283 \pm 2.451}$	73.523 ± 3.343
5	20	$\mathbf{15.052 \pm 1.964}$	677.447 ± 38.226	73.312 ± 2.584	$\mathbf{73.179 \pm 3.551}$
10	5	$\mathbf{21.928 \pm 3.514}$	349.175 ± 23.213	$\mathbf{73.240 \pm 2.603}$	73.609 ± 3.113
10	10	$\mathbf{22.495 \pm 2.961}$	691.875 ± 33.476	73.638 ± 2.712	$\mathbf{73.612 \pm 3.029}$
10	20	$\mathbf{22.766 \pm 3.302}$	1380.407 ± 48.064	73.562 ± 3.019	$\mathbf{73.507 \pm 3.999}$
20	5	$\mathbf{37.660 \pm 4.980}$	693.235 ± 56.103	73.480 ± 2.978	$\mathbf{73.428 \pm 2.966}$
20	10	$\mathbf{39.573 \pm 7.067}$	1380.407 ± 48.064	$\mathbf{73.375 \pm 2.990}$	73.476 ± 3.050
20	20	$\mathbf{42.467 \pm 3.962}$	2763.428 ± 97.766	$\mathbf{73.435 \pm 2.922}$	73.477 ± 3.316

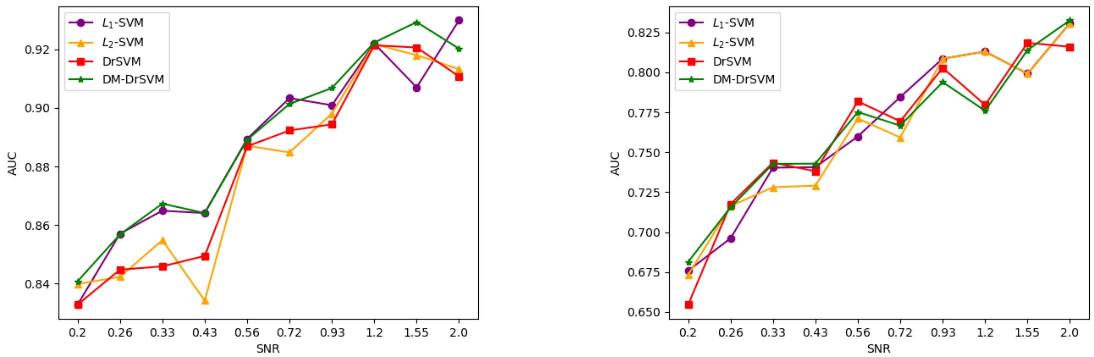
solution path with respect to λ_1. When the number of λ_1 increases, we do not need to recalculate the solution path for the added λ_1.

Table 4.2: Experiments of DM-DrSVM on SCADI Data

Hyperparameter		Computational time		Test AUC Score	
# of λ_2	# of λ_1	SP method (s)	CVXPY(s)	SP method	CVXPY
5	5	$\mathbf{6.858 \pm 0.431}$	44.910 ± 0.739	88.365 ± 7.523	$\mathbf{89.698 \pm 8.242}$
5	10	$\mathbf{6.910 \pm 0.379}$	89.384 ± 2.038	88.365 ± 7.523	$\mathbf{89.643 \pm 8.412}$
5	20	$\mathbf{7.109 \pm 0.398}$	178.662 ± 2.399	88.365 ± 7.523	$\mathbf{89.698 \pm 7.160}$
10	5	$\mathbf{14.306 \pm 0.833}$	91.972 ± 2.474	$\mathbf{91.032 \pm 8.003}$	90.476 ± 8.770
10	10	$\mathbf{14.452 \pm 0.907}$	184.192 ± 3.077	$\mathbf{90.615 \pm 7.849}$	90.476 ± 8.770
10	20	$\mathbf{14.846 \pm 0.829}$	367.418 ± 8.759	$\mathbf{90.615 \pm 7.849}$	89.976 ± 8.345
20	5	$\mathbf{29.226 \pm 1.784}$	185.207 ± 4.357	$\mathbf{90.476 \pm 7.540}$	$\mathbf{90.476 \pm 7.540}$
20	10	$\mathbf{29.773 \pm 1.886}$	371.158 ± 7.003	$\mathbf{90.476 \pm 7.540}$	$\mathbf{90.476 \pm 7.540}$
20	20	$\mathbf{30.552 \pm 1.959}$	742.290 ± 15.842	$\mathbf{90.476 \pm 7.540}$	$\mathbf{90.476 \pm 7.540}$

The results for SCADI data is shown in Table 4.2. SCADI data is high dimensional data with 206 features and 70 data points. SP method is also more efficient than CVXPY solver in SCADI data. High dimensional data is hard to deal with, but SP method also retains good test AUC scores.

Then, we generate synthetic data under different signal-to-noise ratios (SNR). We first generate the coefficient $\boldsymbol{w} \in \mathbb{R}^d$. The input $\boldsymbol{x}_i$ is generated as $\boldsymbol{x}_i \sim N(0, \Sigma)$. The noisy measurements are generated as $\xi_i \sim N(0, \sigma^2 I)$, where σ is defined to meet



(a) Simulated Data (Dim = 5) (b) Simulated Data (dim = 50)

Figure 4.3: Results with Label Noise

the SNR level $SNR = \boldsymbol{w}^\intercal \Sigma \boldsymbol{w}/\sigma^2$. The label for the training data are computed by $y_i = sign(\boldsymbol{w}^\intercal \boldsymbol{x} + \xi_i)$. The label for the test data are generated by $y_i = sign(\boldsymbol{w}^\intercal \boldsymbol{x})$.

In Figure 4.3a, the dimension of the data is 5, the number of the data points is 200. There is no correlation between each feature. When the SNR levels are low, the DM-DrSVM model achieves the best AUC scores among the four models. This is because the DRO formulations considers the label uncertainties. So DM-DrSVM can give robust decisions given noisy labels. As the SNR increases, the DM-DrSVM model still retains better performance than ℓ_2-SVM and DrSVM.

In Figure 4.3b, we increase the dimension to 50 and add correlations between some of the features. DM-DrSVM still performs the best in low SNR settings. It demonstrates good performance in high dimensional data sets.

Next, we apply ℓ_1-SVM, ℓ_2-SVM, DrSVM and DM-DrSVM in real data sets from UCI Machine Learning Repository. The information of the data sets are summarized in Table 4.3. We also calculate variance inflation factor (VIF) to quantify the multi-collinearity (independent variables are highly correlated) in the data. The larger the VIF, the more likely multicollinearity will appear. Each data set is split into 80% for training and 20% for test. The experiments are repeated 10 times and the average results are shown in Table 4.4.

Table 4.3: Information of Real Data Sets

Data set	Number of instances	Number of Features	VIF (max, mean)	Feature Characteristics
Breast cancer	569	32	3806.11, 337.31	Real
Diabetes	768	8	1.59, 1.35	Real, integer
Heart	270	13	1.82, 1.39	Categorical, Real
SCADI	70	206	Inf, Inf	Integer
Ionosphere	351	34	7.11, 3.23	Real, Integer
Sonar	208	60	39.33, 15.71	Real

Table 4.4: AUC Scores of ℓ_1-SVM, ℓ_2-SVM, DrSVM and DM-DrSVM

Dataset	ℓ_1-SVM	ℓ_2-SVM	DrSVM	DM-DrSVM
Breast cancer	92.017 ± 5.361	93.673 ± 4.799	93.757 ± 4.600	$\mathbf{93.932 \pm 4.227}$
Diabetes	71.584 ± 4.807	71.650 ± 5.047	71.720 ± 5.001	$\mathbf{71.787 \pm 4.885}$
Heart	83.685 ± 3.857	83.241 ± 2.899	83.414 ± 2.644	$\mathbf{84.021 \pm 3.773}$
SCADI	90.285 ± 8.283	90.948 ± 7.956	90.893 ± 9.107	$\mathbf{91.448 \pm 8.343}$
Ionosphere	85.192 ± 4.542	84.957 ± 5.419	84.584 ± 4.472	$\mathbf{85.239 \pm 6.419}$
Sonar	73.316 ± 3.522	75.293 ± 3.925	75.627 ± 4.066	$\mathbf{75.711 \pm 4.296}$

As shown in Table 4.4, DM-DrSVM model achieves the best generalization performance in all the data sets. This is because the real data are usually corrupted with noise. DM-DrSVM takes into consideration the noise structure during the DRO formulation, so it is more likely to find the true input-output relationship.

In Breast cancer data set, both the maximum and mean value of VIF is much larger than 10, which means that some of the features are correlated. ℓ_2 regularization term can help to deal with this problem. So in this data set, ℓ_2-SVM is better than ℓ_1-SVM. In DrSVM, the elastic net regularization is the combination of ℓ_1 and ℓ_2 regularization, besides dealing with the correlation problem, it can also perform feature selection, so DrSVM achieve better AUC scores than ℓ_2-SVM. Compared with DrSVM, DM-DrSVM is more robust to noise, so it achieves the best performance.

In Diabetes data set, the correlation of features is weak and the number of features is small, DM-DrSVM is more robust to noise and performs better. Heart data set has more features than Diabetes, we can find that the feature selection plays an important role here and the model with ℓ_1 regularization terms have better AUC scores. DM-DrSVM is also the best model.

In SCADI data set, the infinite VIF indicates that there is perfect correlation between some features. So ℓ_2 regularization performs better. This data set also has high dimensions, DM-DrSVM can perform feature selection and explain the uncertainties in the data, so it achieves the highest AUC scores.

The Ionosphere data has high dimensions and moderate VIF values. So feature selection provided by ℓ_1 regularization contributed more in this data set. In Soar data set, the correlation is much higher, ℓ_2 regularization performs better.

4.5 Conclusions

In this chapter, we derived a new support vector machine model (DM-DrSVM) with double margin and double regularization terms based on distributionally robust optimization theory. A solution path algorithms is proposed for DM-DrSVM model to efficiently obtain the optimal solutions without retraining the model multiple times. The numerical experiments show that the proposed solution path algorithm can highly speed up the hyperparameter tuning process. We generated synthetic data under different SNR levels. Compared with ℓ_1-SVM, ℓ_2-SVM and DrSVM, the DM-DrSVM model achieves better AUC sores in the low SNR scenarios for both low dimensional data and high dimension data. When applied to real data sets, DM-DrSVM outperforms the other models in all the data sets. It performs pretty good when the there is correlation between the features and it. The proposed can also deal with categorical feature. Future research can be extended to kernel space to deal with nonlinear classification problems. Kernel distributionally robust optimization models can be derived and solution path algorithms can be designed for the tuning of the hyperparameters.

Chapter 5

Solution Path Algorithm for Kernel Distributionally Robust Support Vector Machines

In this chapter, we present a kernel distributionally robust support vector machine model, where the synergy of kernel methods and DRO is harnessed to enhance the generalization and robustness of support vector machines. The proposed model is capable of capturing complex, nonlinear decision boundaries, making them applicable to a wide array of real-word scenarios. A solution path algorithm is proposed to efficiently solve this model and speed up the hyperparameter tuning. Numerical experiments show that the solution path algorithm is very efficient and the proposed model demonstrates good generalization performance in different data sets.

The remainder of this chapter is organized as follows. Chapter 5.1 introduces the background of this chapter. In Chapter 5.2, we derive the kernel distributionally robust optimization model and explores its solution property. Chapter 5.3 presents the solution path algorithms for the proposed model. In Chapter 5.4, we perform numerical experiments to test the performance of the proposed model and the solution path algorithms. Finally, Chapter 5.5 concludes this chapter.

5.1 Introduction

Support Vector Machine (SVM) is a powerful supervised learning algorithm for classification and regression analysis. In traditional SVM, the empirical distribution of the training data is used to approximate the true distribution. But in real applications, due to measurement errors, sampling errors, incomplete data and many other problems, the empirical distribution may deviate from the true distribution.

Distributionally robust optimization (DRO) is proposed to address the data uncertainty problem. Instead of considering only the empirical distribution, DRO include a series of distributions centered around the empirical distribution by constructing an ambiguity set. DRO provides robust solutions by minimizing the worst-case expected loss under the constraint of the ambiguity set.

As an attractive tool for improving machine learning models, many new machine learning models are developed based on DRO. Shaeezadeh-Abadeh et al. [8] developed a distributionally robust logistic regression model. Chen and Paschalidis [18] proposed a robust learning approach for regression problems based on DRO. A distributionally robust framework for support vector machines using Kantorovich metric is proposed by Lee and Mehortra [16]. Faccini et al. [126] presented a moment-based distributionally robust support vector machines model.

Existing DRO formulated SVMs are mainly focused on the linear space. In real applications, some of the data are not linearly separable. Kernel methods are often used to map the original input space into a higher dimensional feature space, where the data might become linearly separable. Inspired by the kernel methods, we extend the DRO formulation to nonlinear hypotheses that range over a reproducing kernel Hilbert spaces and propose kernel distributionally robust support vector machines model.

The proposed model can be reformulated as regularized machine learning model, the radius λ of the Wasserstein ambiguity set becomes the regularization hyperparameter. It controls the bias-variance trade-off of and highly influence the performance of the model. Thus, selecting appropriate hyperparameters is very important. Grid search is the most common method for hyperparameter tuning. To obtain the optimal solutions to different hyperparameters, we need to train the models multiple times and this is very inefficient.

Solution path algorithms provide another approach to obtain the optimal solutions with respect to different hyperparameters without retraining the models multiple times. So it can be used to speed up the grid search method. Solution path algorithms have wide applications in support vector machines. Hastie et al. [37] firstly proposed a solution path algorithm to fit the entire of SVM solutions with respect to different hyperparameters. Zhu et al. [41] proposed an efficient algorithm to compute the whole solution path of ℓ_1-SVM. Wang et al. [44] developed the solution path algorithm to track the entire solution path of doubly regularized SVM. The solution path algorithms for ν-SVM was developed by Gu et al. [42, 43].

To speed up the cross validation process, we proved the piecewise linear property of the proposed kernel distributionally robust SVM model and develop a solution path algorithm for this model.

5.2 Kernel Distributionally Robust Support Vector Machine

Denote $s = (\boldsymbol{x}, y)$, $\Xi = \mathbb{R}^d \times \mathbb{R}$, the Wasserstein distance between two distributions $\mathbb{Q}$ and $\mathbb{Q}'$ supported on Ξ is defined as

$$W(\mathbb{Q}, \mathbb{Q}') = \inf_{\Xi} \left\{ \int_{\Xi^2} d(s, s')\Pi(ds, ds') : \begin{array}{l} \Pi \text{ is a joint distribution of } s \text{ and } s' \\ \text{with marginals } \mathbb{Q} \text{ and } \mathbb{Q}' \text{ respectively} \end{array} \right\}$$

where $d(s, s') = \|\boldsymbol{x} - \boldsymbol{x}'\| + \frac{\kappa}{2}|y - y'|$. $\kappa > 0$ is used to measure the relative emphasis between feature mismatch and label uncertainty. The empirical distribution for the training data $\{(\hat{\boldsymbol{x}}_i, \hat{y}_i)\}_{i=1}^n$ is $\hat{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^n \delta_{(\hat{\boldsymbol{x}}_i, \hat{y}_i)}$. Then we can construct the Wasserstein ambiguity set around $\hat{\mathbb{P}}$

$$\mathcal{B}_\lambda(\hat{\mathbb{P}}) = \{\mathbb{Q} : W(\mathbb{Q}, \hat{\mathbb{P}}) \leq \lambda\}$$

The distributionally robust support vector machine over nonlinear hypothesis can be formulated as

$$\min_{h\in\mathbb{H}} \sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}})} \mathbb{E}^{\mathbb{Q}}[\ell(h(\boldsymbol{x}), y)], \tag{5.1}$$

Problem (5.1) is difficult to solve due the nonlinearity of the hypothesis $h \in \mathbb{H}$. Here, we replace the $\boldsymbol{x}$ with $\boldsymbol{x}_{\mathbb{H}} \in \mathbb{H}$ and h with $h_{\mathbb{H}} \in \mathbb{H}$. Then we obtained the following lifted learning problem

$$\min_{h\in\mathbb{H}} \sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}}^{\mathbb{H}})} \mathbb{E}^{\mathbb{Q}}[\ell(\langle h, \boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}}, y)], \tag{5.2}$$

According to Theorem 26 in [8], if the kernel satisfies the calmness condition, (5.2) provides an upper bound on (5.1).

Theorem 5.1. Suppose $\mathbb{X} \in \mathbb{R}^n, \mathbb{Y} = \{+1, -1\}$. If loss function ℓ is hinge loss $\ell(\boldsymbol{x}, y) = 1 - yh(\boldsymbol{x})$, then (5.2) can be reformulated as

$$\min_{h\in\mathbb{H},\rho,\xi_i} \quad \lambda\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{5.3a}$$

$$s.t.\ 1 - y_i h(\boldsymbol{x}_i) \le \xi_i, \quad i = 1, \dots, n \tag{5.3b}$$

$$1 + y_i h(\boldsymbol{x}_i) - \kappa\rho \le \xi_i, \quad i = 1, \dots, n \tag{5.3c}$$

$$\|h\|_{\mathbb{H}} \le \rho \tag{5.3d}$$

$$\xi_i \ge 0, \quad i = 1, \dots, n \tag{5.3e}$$

Proof. Let $s = (\langle h, \boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}}, y)$, By the definition of Wasserstein distance, we can obtain

$$\sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}}^{\mathbb{H}})} \mathbb{E}^{\mathbb{Q}}[\ell(s)] = \begin{cases} \sup_{\Pi} & \int_{\Xi^2} \ell(s)\Pi(ds, ds') \\ s.t. & \int_{\Xi^2} d(s, s')\Pi(ds, ds') \le \lambda \end{cases}$$

Since the marginal distribution of s' is the uniform distribution on the training samples, we have

$$\Pi(ds, ds') = \frac{1}{n}\sum_{i=1}^{n} \delta_{\hat{s}_i} d(s')\mathbb{Q}_i(ds)$$

So we can obtain

$$\sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}}^{\mathbb{H}})} \mathbb{E}^{\mathbb{Q}}[\ell(s)] = \begin{cases} \sup_{\Pi} & \frac{1}{n}\sum_{i=1}^n \int_\Xi \ell(s)\mathbb{Q}_i(ds) \\ s.t. & \frac{1}{n}\sum_{i=1}^n \int_\Xi d(s,\hat{s}_i)\mathbb{Q}_i(ds) \le \lambda \\ & \int_\Xi \mathbb{Q}_i(ds) = 1, i = 1,\dots,n \end{cases}$$

Let $\rho \geq 0$ be the Lagrange multiplier, we can obtain

$$\begin{aligned} \sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}}^{\mathbb{H}})} \mathbb{E}^{\mathbb{Q}}[\ell(s)] &= \sup_{\mathbb{Q}_i}\inf_{\rho} \frac{1}{n}\sum_{i=1}^n \int_\Xi \ell(s)\mathbb{Q}_i(ds) + \rho(\lambda - \frac{1}{n}\sum_{i=1}^n \int_\Xi d(s,\hat{s}_i)\mathbb{Q}_i(ds)) \\ &\le \inf_{\rho}\sup_{\mathbb{Q}_i}\ \lambda\rho + \frac{1}{n}\sum_{i=1}^n \int_\Xi (\ell(s) - \rho d(s,\hat{s}_i))\mathbb{Q}_i(ds) \\ &= \inf_{\rho}\ \lambda\rho + \sup_{s\in\Xi}\ \frac{1}{n}\sum_{i=1}^n (\ell(s) - \rho d(s,\hat{s}_i)) \end{aligned}$$

where the second inequality follows from the max-min inequality. The last equality is because $\mathcal{M}(\Xi)$ contains all the Dirac distributions supported on Ξ. Let $\xi_i (i = 1,\dots,n)$ be the epigraphical auxiliary variables, then we can obtain

$$\begin{aligned} &\begin{cases} \inf_{\rho,\xi_i} & \lambda\rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. & \sup_{s\in\Xi}\ \ell(s) - \rho d(s,\hat{s}_i) \le \xi_i, i = 1,\dots,n \end{cases} \\ = &\begin{cases} \inf_{\rho,\xi_i} & \lambda\rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. & \sup_{(\boldsymbol{x}_{\mathbb{H}},y)\in\mathbb{H}\times\mathbb{Y}} \ell(\langle h,\boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}},y) - \rho(\|\boldsymbol{x}_{\mathbb{H}} - \hat{\boldsymbol{x}}_i\| + \frac{\kappa}{2}|y-\hat{y}_i|) \le \xi_i, i = 1,\dots,n \end{cases} \\ = &\begin{cases} \inf_{\rho,\xi_i} & \lambda\rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. & \sup_{\boldsymbol{x}_{\mathbb{H}}\in\mathbb{H}} L(\hat{y}_i\langle h,\boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}}) - \rho\|\boldsymbol{x}_{\mathbb{H}} - \hat{\boldsymbol{x}}_i\| \le \xi_i, \quad i = 1,\dots,n \\ & \sup_{\boldsymbol{x}_{\mathbb{H}}\in\mathbb{H}} L(-\hat{y}_i\langle h,\boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}}) - \rho\|\boldsymbol{x}_{\mathbb{H}} - \hat{\boldsymbol{x}}_i\| - \kappa\rho \le \xi_i, \quad i = 1,\dots,n \end{cases} \\ = &\begin{cases} \inf_{\rho,\xi_i} & \lambda\rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. & L(\hat{y}_i\langle h,\hat{\boldsymbol{x}}_i\rangle) \le \xi_i, i \in [n] \\ & L(-\hat{y}_i\langle h,\hat{\boldsymbol{x}}_i\rangle) - \kappa\rho \le \xi_i, \quad i = 1,\dots,n \\ & Lip(L)\|h\|_{\mathbb{H}} \le \rho \end{cases} \end{aligned}$$

where the last equality is based on Lemma 47 in [8]. For the hinge loss $L(\hat{y}_i\langle h,\hat{\boldsymbol{x}}_i\rangle) = 1 - \hat{y}_i h(\hat{\boldsymbol{x}}_i)$, we have $Lip(L) = 1$. Then we can obtain

$$\min_{h\in\mathbb{H}}\sup_{\mathbb{Q}\in\mathcal{B}_\lambda(\hat{\mathbb{P}}^{\mathbb{H}})} \mathbb{E}^{\mathbb{Q}}[\ell(\langle h,\boldsymbol{x}_{\mathbb{H}}\rangle_{\mathbb{H}},y)] = \begin{cases} \min_{h\in\mathbb{H},\rho,\xi_i} & \lambda\rho + \frac{1}{n}\sum_{i=1}^n \xi_i \\ s.t. & 1 - \hat{y}_i h(\hat{\boldsymbol{x}}_i) \le \xi_i, \quad i = 1,\dots,n \\ & 1 + \hat{y}_i h(\hat{\boldsymbol{x}}_i) - \kappa\rho \le \xi_i, \quad i = 1,\dots,n \\ & \|h\|_{\mathbb{H}} \le \rho \end{cases}$$

Thus, the kernel distributionally robust support vector machines can be written as

$$\min_{h\in\mathbb{H},\rho,\xi_i} \ \lambda\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{5.4a}$$

$$s.t.\ 1 - y_i h(\boldsymbol{x}_i) \le \xi_i, \quad i = 1,\dots,n \tag{5.4b}$$

$$1 + y_i h(\boldsymbol{x}_i) - \kappa\rho \le \xi_i, \quad i = 1,\dots,n \tag{5.4c}$$

$$\|h\|_{\mathbb{H}} \le \rho \tag{5.4d}$$

$$\xi_i \ge 0, \quad i = 1,\dots,n \tag{5.4e}$$

□

Different from the traditionally kernel SVM, the DRO formulation introduced another margin corresponding to constraint (5.4d). Thus the newly proposed model is named as Double Margin Kernel Support Vector Machine (DM-KSVM). DM-KSVM can also be written as

$$\begin{aligned}
\min_{\boldsymbol{w},\rho,\xi_i} \ & \lambda\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\
s.t.\ & 1 - y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) \le \xi_i, \quad i = 1,\dots,n \\
& 1 + y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \kappa\rho \le \xi_i, \quad i = 1,\dots,n \\
& \frac{1}{2}\|\boldsymbol{w}\|_2^2 \le \rho \\
& \xi_i \ge 0, \quad i = 1,\dots,n
\end{aligned}$$

To facilitate the calculation, we approximate $\kappa\rho$ to be κ and assume $\frac{1}{2}\|\boldsymbol{w}\|_2^2 \le \rho$ is satisfied, then we can obtain

$$\begin{aligned}
\min_{\boldsymbol{w},\rho,\xi_i} \ & \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\
s.t.\ & 1 - y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) \le \xi_i, \quad i = 1,\dots,n
\end{aligned}$$

$$1 + y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \kappa \leq \xi_i, \quad i = 1, \dots, n$$
$$\xi_i \geq 0, \quad i = 1, \dots, n$$

The reformulated model retains double margin and is more computationally efficient. The Lagrangian function is

$$\begin{aligned} L(\boldsymbol{w}, \rho, \xi_i, \alpha, \mu, \theta, \eta) =& \lambda \boldsymbol{w} + \frac{1}{n}\sum_{i=1}^{n} \xi_i + \sum_{i=1}^{n} \alpha_i(1 - y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \xi_i) \\ &+ \sum_{i=1}^{n} \mu_i(1 + y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \kappa - \xi_i) - \sum_{i=1}^{n} \eta_i \xi_i \end{aligned}$$

Define $\mathcal{V} = \{j : \boldsymbol{w}_j \neq 0\}$, the Karush–Kuhn–Tucker (KKT) conditions can be written as

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = \lambda \boldsymbol{w}_j - \sum_{i=1}^{n} \alpha_i y_i \phi(\boldsymbol{x}_{ij}) + \mu_i y_i \phi(\boldsymbol{x}_{ij}) = 0, \quad j \in \mathcal{V}$$
$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} -\alpha_i y_i + \mu_i y_i = 0$$
$$\frac{\partial L}{\partial \xi_i} = \frac{1}{n} - \alpha_i - \mu_i - \eta_i = 0, \quad i = 1, \dots, n$$
$$\alpha_i(1 - y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \xi_i) = 0, \quad i = 1, \dots, n$$
$$\mu_i(1 + y_i(\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\rangle + b) - \kappa - \xi_i) = 0, \quad i = 1, \dots, n$$
$$\eta_i \xi_i = 0, \quad i = 1, \dots, n$$

Then we have

$$\boldsymbol{w} = \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i \phi(\boldsymbol{x}_i)$$

So the decision function is given by

$$h(\boldsymbol{x}) = \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

Based on KKT conditions, we define the following sets

$$\mathcal{L} = \{i : y_i h(\boldsymbol{x}_i) < 1\}, \quad \alpha_i = \frac{1}{n}, \mu_i = 0$$

$$\mathcal{E}_L = \{i : y_i h(\boldsymbol{x}_i) = 1\}, \quad 0 \leq \alpha_i \leq \frac{1}{n}, \mu_i = 0$$
$$\mathcal{E}_M = \{i : 1 < y_i h(\boldsymbol{x}_i) < \kappa\rho - 1\}, \quad \alpha_i = \mu_i = 0$$
$$\mathcal{E}_R = \{i : y_i h(\boldsymbol{x}_i) = \kappa\rho - 1\}, \quad \alpha_i = 0, 0 \leq \mu_i \leq \frac{1}{n}$$
$$\mathcal{R} = \{i : y_i h(\boldsymbol{x}_i) > \kappa\rho - 1\}, \quad \mu_i = \frac{1}{n}, \alpha_i = 0$$

For the points in $\mathcal{E}_L$ and $\mathcal{E}_R$, we obtain the system of equations

$$\sum_{i=1}^{n} \alpha_i y_i - \mu_i y_i = 0$$
$$y_i(\frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b) = 1, \quad i \in \mathcal{E}_L$$
$$y_i(\frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b) = \kappa - 1, \quad i \in \mathcal{E}_R$$

Define $\alpha_0 = \lambda b$, we can obtain

$$\sum_{i=1}^{n} \alpha_i y_i - \mu_i y_i = 0$$
$$y_i(\sum_{j=1}^{n}(\alpha_j - \mu_j) y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \alpha_0) = \lambda, \quad i \in \mathcal{E}_L$$
$$y_i(\sum_{j=1}^{n}(\alpha_j - \mu_j) y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \alpha_0) = (\kappa - 1)\lambda, \quad i \in \mathcal{E}_L$$

Take derivative with respect to λ, then we can obtain

$$\sum_{i \in \mathcal{E}_L} \frac{\Delta\alpha_i}{\Delta\lambda} y_i - \sum_{i \in \mathcal{E}_R} \frac{\Delta\mu_i}{\Delta\lambda} y_i = 0$$
$$\sum_{j \in \mathcal{E}_L} \frac{\Delta\alpha_j}{\Delta\lambda} y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{j \in \mathcal{E}_R} \frac{\Delta\mu_j}{\Delta\lambda} y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{\Delta\alpha_0}{\Delta\lambda} = y_i, \quad i \in \mathcal{E}_L$$
$$\sum_{j \in \mathcal{E}_L} \frac{\Delta\alpha_j}{\Delta\lambda} y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{j \in \mathcal{E}_R} \frac{\Delta\mu_j}{\Delta\lambda} y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{\Delta\alpha_0}{\Delta\lambda} = (\kappa - 1) y_i, \quad i \in \mathcal{E}_R$$

The above systems of linear equations have $|\mathcal{E}_L| + |\mathcal{E}_R| + 1$ equations and $|\mathcal{E}_L| + |\mathcal{E}_R| + 1$ unknown variables. If there is no singular problem, it will produce a unique solution. The solution of this system does not contain λ, which means that the gradients of the variables are constant. As long as $\mathcal{E}_L$ and $\mathcal{E}_R$ do not change, the variables will change linearly with respect to λ. If $\mathcal{E}_L$ and $\mathcal{E}_R$ change, we will turn to another linear piece. So the optimal solutions will change piecewise linearly with respect to λ.

5.3 Solution Path Algorithms for DM-KSVM

The λ is initialized to be $+\infty$, then $\rho = 0$, the problem is reduced to

$$\begin{aligned}
\min_{\xi_i, b} \ & \frac{1}{n}\sum_{i=1}^{n}\xi_i \\
s.t. \ & 1 - y_i b \leq \xi_i, \quad i = 1, \ldots, n \\
& 1 + y_i b - \kappa \leq \xi_i, \quad i = 1, \ldots, n \\
& \xi_i \geq 0, \quad i = 1, \ldots, n
\end{aligned}$$

Define $\mathcal{I}_+ = i : y_i = 1$ and $\mathcal{I}_- = i : y_i = -1$, then we have $n_+ = |\mathcal{I}_+|, n_- = |\mathcal{I}_-|$. As λ decreases, if $n_+ = n_-$, two points i_+ and i_- will join the $\mathcal{E}_L$ simultaneously. The index can be obtained by $i_+ = \arg\max_{i \in \mathcal{I}_+} h(\boldsymbol{x}_i)$ and $i_- = \arg\min_{i \in \mathcal{I}_-} h(\boldsymbol{x}_i)$ Then we can obtain

$$\begin{cases} \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_+}, \boldsymbol{x}_i) + b = 1 \\ \frac{1}{\lambda}\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_-}, \boldsymbol{x}_i) + b = -1 \end{cases}$$
$$\Rightarrow \begin{cases} \lambda = \frac{\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_+}, \boldsymbol{x}_i) - \sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_-}, \boldsymbol{x}_i)}{2} \\ b = -\frac{\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_+}, \boldsymbol{x}_i) + \sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_-}, \boldsymbol{x}_i)}{\sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_+}, \boldsymbol{x}_i) - \sum_{i=1}^{n}(\alpha_i - \mu_i) y_i K(\boldsymbol{x}_{i_-}, \boldsymbol{x}_i)} \end{cases}$$

If $n_+ > n_-$, the dual formulation can be written as

$$\max_{\alpha, \mu} \ \sum_{i=1}^{n}(\alpha_i + (1 - \kappa)\mu_i) - \frac{1}{2\lambda}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \mu_i)(\alpha_j - \mu_j) y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{5.5a}$$

$$s.t. \ \sum_{i=1}^{n}(\alpha_i - \mu_i) y_i = 0 \tag{5.5b}$$

$$0 \leq \alpha_i \leq \frac{1}{n}, \quad i = 1, \ldots, n \tag{5.5c}$$

$$0 \leq \mu_i \leq \frac{1}{n}, \quad i = 1, \ldots, n \tag{5.5d}$$

If $b = 1$, we have $h(\boldsymbol{x}) = 1$. Since $yh(\boldsymbol{x}) \leq 1$, we have $\mu_i = 0 (i = 1, \ldots, n)$. Since $h(\boldsymbol{x}) = 1$, the points in $\mathcal{I}_-$ will be misclassified, so the corresponding α_i is $1/n$. Then constraint (5.5c) can be written as

$$\sum_{i=1}^{n} \alpha_i y_i = \sum_{i \in \mathcal{I}_+} \alpha_i - \sum_{i \in \mathcal{I}_-} \alpha_i = 0$$

Since $\sum_{i \in \mathcal{I}_-} \alpha_i = n_-/n$, then th constraint can be written as

$$\sum_{i \in \mathcal{I}_+} \alpha_i = \frac{n_-}{n}$$

Then we can obtain the initial values by solving the following problem

$$\begin{aligned}
\min_{\alpha} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
s.t. \quad & \sum_{i \in \mathcal{I}_+} \alpha_i = \frac{n_-}{n} \\
& \alpha_i = \frac{1}{n}, \quad i \in \mathcal{I}_- \\
& 0 \leq \alpha_i \leq \frac{1}{n}, \quad i \in \mathcal{I}_+
\end{aligned}$$

Next, we need to identify the events when $\mathcal{E}_L$ and $\mathcal{E}_R$ change. Denote

$$h(\boldsymbol{x}) = \frac{1}{\lambda} \Big(\sum_{j=1}^{n} (\alpha_j - \mu_j) y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + \alpha_0 \Big)$$

Let $\lambda = \lambda^t + \Delta\lambda$, we have

$$h(\boldsymbol{x}) = \Big[h(\boldsymbol{x}) - \frac{\lambda^t}{\lambda} h^t(\boldsymbol{x}) \Big] + \frac{\lambda^t}{\lambda} h^t(\boldsymbol{x})$$

$$= \Big[\frac{1}{\lambda}\Big(\sum_{j=1}^{n}(\alpha_j - \mu_j)y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + \alpha_0\Big) - \frac{\lambda^t}{\lambda}\frac{1}{\lambda^t}\Big(\sum_{j=1}^{n}(\alpha_j^t - \mu_j^t)y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + \alpha_0^t\Big)\Big] + \frac{\lambda^t}{\lambda}h^t(\boldsymbol{x})$$

$$= \frac{1}{\lambda}\Big[\sum_{j\in\mathcal{E}_L}(\alpha_j - \alpha_j^t)y_j K(\boldsymbol{x}, \boldsymbol{x}_j) - \sum_{j\in\mathcal{E}_R}(\mu_j - \mu_j^t)y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + (\alpha_0 - \alpha_0^t) + \lambda^t h^t(\boldsymbol{x})\Big]$$

Let $\Delta\alpha_j = \alpha_j - \alpha_j^t, \Delta\mu_j = \mu_j - \mu_j^t, \Delta\alpha_0 = \alpha_0 - \alpha_0^t$, we can obtain

$$\sum_{j\in\mathcal{E}_L}\Delta\alpha_j y_j K(\boldsymbol{x}, \boldsymbol{x}_j) - \sum_{j\in\mathcal{E}_R}\Delta\mu_j y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + \Delta\alpha_0 + \lambda^t h^t(\boldsymbol{x}) = (\lambda^t + \Delta\lambda)h(\boldsymbol{x})$$

Denote $f^t(\boldsymbol{x}) = \sum_{j\in\mathcal{E}_L}\frac{\Delta\alpha_j}{\Delta\lambda}y_j K(\boldsymbol{x}, \boldsymbol{x}_j) - \sum_{j\in\mathcal{E}_R}\frac{\Delta\mu_j}{\Delta\lambda}y_j K(\boldsymbol{x}, \boldsymbol{x}_j) + \frac{\Delta\alpha_0}{\Delta\lambda}$, we have

$$\Delta\lambda f^t(\boldsymbol{x}) + \lambda^t h^t(\boldsymbol{x}) = \lambda h(\boldsymbol{x})$$

If the point enters $\mathcal{E}_L$, we have $h(\boldsymbol{x}_i) = y_i$

$$\Delta\lambda f^t(\boldsymbol{x}_i) + \lambda^t h^t(\boldsymbol{x}_i) = (\lambda^t + \Delta\lambda)y_i, \quad i \in \mathcal{E}_L$$

Then we can obtain

$$\Delta\lambda = \lambda^t\frac{y_i - h^t(\boldsymbol{x}_i)}{f^t(\boldsymbol{x}_i) - y_i}, \quad i \in \mathcal{E}_L$$

Similarly, if the point enters $\mathcal{E}_R$, we have $h(\boldsymbol{x}_i) = (\kappa - 1)y_i$, then

$$\Delta\lambda = \lambda^t\frac{(\kappa - 1)y_i - h^t(\boldsymbol{x}_i)}{f^t(\boldsymbol{x}_i) - (\kappa - 1)y_i}, \quad i \in \mathcal{E}_R$$

To facility the calculation of δ_λ, we define a function $g(\delta^-, \delta^+)$

$$g(\delta^-, \delta^+) = \begin{cases} \min\{\delta^-, \delta^+\}, \text{ if } \delta^- \geq 0 \text{ and } \delta^+ \geq 0 \\ +\infty, \text{ if } \delta^- < 0 \text{ and } \delta^+ < 0 \\ \max\{\delta^-, \delta^+\}, \text{ otherwise} \end{cases}$$

Then we have the following events

(1) a point leaves $\mathcal{E}_L$

$$\delta_\lambda^1 = \max_{i \in \mathcal{E}_L} g_i(\delta_i^-, \delta_i^+) \tag{5.6}$$

where $\delta_i^- = \frac{0-\alpha_i}{\Delta\alpha_i/\Delta\lambda}$, $\delta_i^+ = \frac{1/n-\alpha_i}{\Delta\alpha_i/\Delta\lambda}$

(2) a point joins $\mathcal{E}_L$ from $\mathcal{L}$ or $\mathcal{E}_M$

$$\delta_\lambda^2 = \max_{i \in (\mathcal{L} \cup \mathcal{E}_M)_+} \left\{ \lambda^t \frac{y_i - h^t(\boldsymbol{x}_i)}{f^t(\boldsymbol{x}_i) - y_i} \right\}, \tag{5.7}$$

(3) a point leaves $\mathcal{E}_R$

$$\delta_\lambda^3 = \max_{i \in \mathcal{E}_R} g_i(\delta_i^-, \delta_i^+), \tag{5.8}$$

where $\delta_i^- = \frac{0-\mu_i}{\Delta\mu_i/\Delta\lambda}$, $\delta_i^+ = \frac{1/n-\mu_i}{\Delta\mu_i/\Delta\lambda}$

(4) a point moves from $\mathcal{R}$ or $\mathcal{E}_M$ to $\mathcal{E}_R$

$$\delta_\lambda^4 = \max_{i \in (\mathcal{R} \cup \mathcal{E}_M)_+} \left\{ \lambda^t \frac{(\kappa-1)y_i - h^t(\boldsymbol{x}_i)}{f^t(\boldsymbol{x}_i) - (\kappa-1)y_i} \right\}, \tag{5.9}$$

(5) λ reaches to lower bound:

$$\delta_\lambda^5 = \lambda^{min} - \lambda \tag{5.10}$$

Then the solution path algorithm with respect to λ is shown in Algorithm 5.1.

Algorithm 5.1: Solution path algorithm with respect to λ

Initialize λ, α, μ, α_0, $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$

while $\lambda \leq \lambda_{max}$ **do**

  Compute $\delta_\lambda^1, \delta_\lambda^2, \delta_\lambda^3, \delta_\lambda^4, \delta_\lambda^5$ by equations (5.6)-(5.10)

  Find $\delta_{\lambda_1} = \min\{\delta_\lambda^1, \delta_\lambda^2, \delta_\lambda^3, \delta_\lambda^4, \delta_\lambda^5\}$;

  Update $\alpha_i, \alpha_0, \mu_i$ by

$$\alpha_i \leftarrow \alpha_i + \delta_\lambda \frac{\Delta\alpha_i}{\Delta\lambda}, \quad i \in \mathcal{E}_L, \qquad \alpha_0 \leftarrow \alpha_0 + \delta_\lambda \frac{\Delta\alpha_0}{\Delta\lambda}$$
$$\mu_i \leftarrow \mu_i + \delta_\lambda \frac{\Delta\mu_i}{\Delta\lambda}, \quad i \in \mathcal{E}_R$$

  Update $\mathcal{L}$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_R$, $\mathcal{R}$ $\lambda_1 \leftarrow \lambda_1 + \delta_{\lambda_1}$

5.4 Numerical Experiment

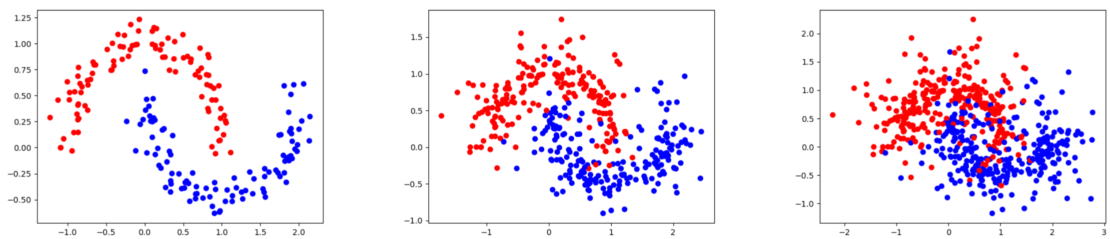
5.4.1 Simulation experiments

The experiments were performed on a Linux server with AMD Zen2 10 core processors (2.3GHz) and 50GB RAM. Two moon data sets are used to test the computational efficiency of the solution path algorithm. They are generated by scikit-learn [127] with different sample size. The solution path algorithm and Gurobi solver are applied to perform the grid search. 5-fold cross validation is employed to select the appropriate hyperparameters. The computation time and test AUC scores were recorded for comparison. The experiment results are shown in Table 5.1.

Table 5.1: Experiments of DM-KSVM on Two Moon Data

# of Samples	# of λ	Computational time		Test AUC Score	
		SP method (s)	Gurobi(s)	SP method	Gurobi
100	5	$\mathbf{2.089 \pm 0.167}$	5.088 ± 0.111	81.736 ± 9.446	$\mathbf{82.317 \pm 7.982}$
	10	$\mathbf{2.667 \pm 0.205}$	9.096 ± 0.166	$\mathbf{82.872 \pm 7.558}$	82.771 ± 7.875
	20	$\mathbf{3.913 \pm 0.343}$	17.137 ± 0.216	$\mathbf{82.872 \pm 7.558}$	82.317 ± 7.982
200	5	$\mathbf{8.289 \pm 0.386}$	20.136 ± 0.332	82.842 ± 8.178	$\mathbf{84.200 \pm 4.647}$
	10	$\mathbf{9.815 \pm 1.828}$	35.987 ± 0.377	$\mathbf{84.786 \pm 4.413}$	82.542 ± 7.336
	20	$\mathbf{13.026 \pm 0.944}$	67.298 ± 0.543	$\mathbf{84.786 \pm 4.413}$	84.417 ± 4.314
500	5	$\mathbf{42.961 \pm 2.208}$	129.763 ± 3.609	84.967 ± 2.261	$\mathbf{85.005 \pm 2.513}$
	10	$\mathbf{49.282 \pm 2.283}$	236.502 ± 3.687	$\mathbf{85.156 \pm 2.419}$	85.127 ± 2.371
	20	$\mathbf{62.809 \pm 4.760}$	447.974 ± 3.572	$\mathbf{85.127 \pm 2.372}$	$\mathbf{85.127 \pm 2.371}$

From Table 5.1, we can find that the solution path algorithm is much faster than directly applying the Gurobi solver. This is because in solution path algorithm, only the points in the active sets are involved in the calculation, which saves lots of time. The computational time of Gurobi solver scale in the total number of programs we need to solve. So when the number of hyperparameters doubles, the computational time of Gurobi solver will double. However, in solution path algorithms, since we already find the entire solution path, the increased time is only caused by more iterations.



(a) Std = 0.1 (b) Std = 0.3 (c) Std = 0.5

Figure 5.1: Two Moon Data Sets with Different Noise

Compared with the Gurobi solver, solution path algorithm also retains good test AUC scores, which means it can select the appropriate hyperparameter.

To test the performance of the proposed model under noisy data, we add Gaussian noise with different standard deviation (range from 0.1 to 0.5) to the two moon data. As shown in Figure 5.1, when the standard deviation increases, the data will become more noisy and is hard to classify. We compared the performance of SVM, kernel SVM, Double Margin SVM (DM-SVM) and DM-KSVM in these data. The results are shown in Figure 5.2. As the noise increases, the AUC scores of all the four models have decreasing trend. When the noise level is low, KSVM achieves the highest AUC scores. As the noise level increases, DM-KSVM obtains the best performance. The DRO formulations consider uncertainties in the data, thus DM-KSVM is more robust in noisy data.

5.4.2 Real data experiments

Next, we compare the performance of the above four models in different real data sets. The information of the data sets are summarized in Table 5.2. We randomly select 80 % for training and 20% for test. The average results over 10 independent trials are shown in Table 5.3. We can observe that DM-KSVM achieves the best

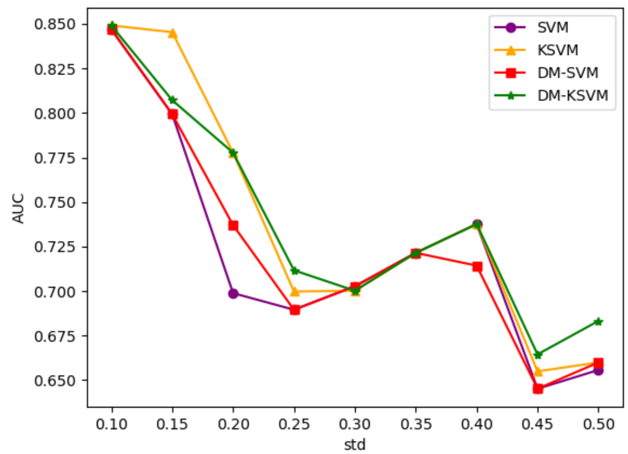


Figure 5.2: Simulation Experiments with Noisy Data

generalization performance among different data sets.

Table 5.2: Information of Data Sets

Data set	Number of instances	Number of Features	Feature Characteristics
Iris	100	4	Real
Diabetes	768	8	Real, Integer
Ionosphere	351	34	Real, Integer
Fourclass	862	2	Integer

Table 5.3: AUC Scores of SVM, KSVM, DM-SVM and DM-KSVM

Dataset	SVM	KSVM	DM-SVM	DM-KSVM
Iris	91.254 ± 5.956	92.452 ± 5.592	92.103 ± 4.930	$\mathbf{92.907 \pm 6.106}$
Diabetes	70.921 ± 2.480	$\mathbf{71.478 \pm 2.524}$	71.346 ± 2.754	71.398 ± 2.973
Ionosphere	80.004 ± 3.951	92.939 ± 2.676	80.121 ± 4.030	$\mathbf{93.760 \pm 2.450}$
Fourclass	70.654 ± 2.332	$\mathbf{70.980 \pm 1.955}$	7.798 ± 2.493	$\mathbf{70.980 \pm 1.955}$

5.5 Conclusions

In this chapter, we propose a novel DM-KSVM model to address the data uncertainty problem in classification. By leveraging the power of both kernel methods and

DRO, DM-KSVM offers a principled approach to enhancing the robustness and generalization of SVM in real-world applications. An efficient solution path algorithm is introduced to speed up the hyperparameter tuning. The experimental results demonstrated the superiority of the solution path algorithm compared to existing methods. DM-KSVM demonstrated improved classification accuracy and robustness on various benchmark data sets. By addressing the challenges of model robustness and uncertainty, it contributes to the development of more reliable and adaptable machine learning models. The proposed model can deal with the correlation between features and categorical feature. Moreover, the computational efficiency of solution path algorithm makes it a viable tool for large scale problems, further broadening its practical utility. Future work could explore DRO formulations for other learning paradigms and develop solution path algorithm to tune other hyperparameters, ultimately advancing the understanding and implementation of robust machine learning techniques.

Chapter 6
Conclusions and Future Work

In this dissertation, we mainly focused on distributionally robust regression and classification problems, and presented novel solution path algorithms to efficiently solve the proposed model. In this chapter, we first summarize the contributions of the dissertation, and then we discuss some future research directions.

Conclusions. In this dissertation, we developed DRO formulated regression and classification models, and proposed solution path algorithms tailored for improvement of computational efficiency. We introduced novel formulations that explicitly account for the data uncertainty, enabling models to generalize well to new and unseen data. The proposed solution path algorithms are based on parametric optimization theory, highly improving the computational efficiency. Empirical experiments conducted on simulated data sets and benchmark real data sets demonstrated the accuracy and robustness of the proposed DRO formulated model and the efficiency of the solution path algorithms.

Future Research Directions. For future research, there are several directions that can be extended based on the work of this dissertation:

- Extending the DRO model with different ambiguity sets. Besides the Wasserstein ambiguity sets, there are many other ambiguity sets, such as Kullback-Leibler divergence ambiguity set, Sinkhorn ambiguity and Moment-based ambiguity set.

- Developing solution path algorithms for other hyperparameters. The solution path algorithm developed in Chapter 5 is mainly for the regularization hyperparameter. For the kernel hyperparameter, its solution path is piecewise smooth not piecewise linear, it could be interesting to develop solution path algorithms to trace the the solution path with respect to kernel hyperparameter.

- Combining DRO formulation technique with more machine learning tasks. In this dissertation, we mainly focus on supervised learning tasks, such as regression and classification. The future work may consider to apply DRO to semi-supervised learning and reinforcement learning tasks. Moreover, applying DRO formulation technique to neural network is also a promising research direction.

- Incorporating more advanced regularization term. ℓ_1-norm and ℓ_2-norm regularization are mainly investigated in this dissertation. It would be beneficial to develop solution path algorithms for other regularization terms, such as Group Lasso, $\ell_p(0 < p < 1)$ regularization term.

References

[1] T. B. Trafalis and R. C. Gilbert, "Robust classification and regression using support vector machines," *European Journal of Operational Research*, vol. 173, no. 3, pp. 893–909, 2006.

[2] N. Fan, E. Sadeghi, and P. M. Pardalos, "Robust support vector machines with polyhedral uncertainty of the input data," in *Learning and Intelligent Optimization: 8th International Conference, Lion 8, Gainesville, FL, USA, February 16-21, 2014. Revised Selected Papers 8*, pp. 291–305, Springer, 2014.

[3] A. Ben-Tal, S. Bhadra, C. Bhattacharyya, and J. Saketha Nath, "Chance constrained uncertain classification via robust optimization," *Mathematical programming*, vol. 127, pp. 145–173, 2011.

[4] X. Wang, N. Fan, and P. M. Pardalos, "Robust chance-constrained support vector machines with second-order moment information," *Annals of Operations Research*, vol. 263, pp. 45–68, 2018.

[5] G. C. Calafiore and L. E. Ghaoui, "On distributionally robust chance-constrained linear programs," *Journal of Optimization Theory and Applications*, vol. 130, pp. 1–22, 2006.

[6] E. Delage and Y. Ye, "Distributionally robust optimization under moment uncertainty with application to data-driven problems," *Operations research*, vol. 58, no. 3, pp. 595–612, 2010.

[7] Z. Wang, P. W. Glynn, and Y. Ye, "Likelihood robust optimization for data-driven problems," *Computational Management Science*, vol. 13, pp. 241–261, 2016.

[8] S. Shafieezadeh-Abadeh, D. Kuhn, and P. M. Esfahani, "Regularization via mass transportation," *Journal of Machine Learning Research*, vol. 20, no. 103, pp. 1–68, 2019.

[9] A. Ben-Tal, D. Den Hertog, A. De Waegenaere, B. Melenberg, and G. Rennen, "Robust solutions of optimization problems affected by uncertain probabilities," *Management Science*, vol. 59, no. 2, pp. 341–357, 2013.

[10] R. Gao and A. Kleywegt, "Distributionally robust stochastic optimization with wasserstein distance," *Mathematics of Operations Research*, vol. 48, no. 2, pp. 603–655, 2023.

[11] P. Mohajerin Esfahani and D. Kuhn, "Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations," *Mathematical Programming*, vol. 171, no. 1, pp. 115–166, 2018.

[12] J. Blanchet and K. Murthy, "Quantifying distributional model risk via optimal transport," *Mathematics of Operations Research*, vol. 44, no. 2, pp. 565–600, 2019.

[13] J. Wang, R. Gao, and Y. Xie, "Sinkhorn distributionally robust optimization," *arXiv preprint arXiv:2109.11926*, 2021.

[14] N. Fournier and A. Guillin, "On the rate of convergence in wasserstein distance of the empirical measure," *Probability theory and related fields*, vol. 162, no. 3-4, pp. 707–738, 2015.

[15] S. Shafieezadeh Abadeh, P. M. Mohajerin Esfahani, and D. Kuhn, "Distributionally robust logistic regression," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[16] C. Lee and S. Mehrotra, "A distributionally-robust approach for finding support vector machines," *Available from Optimization Online*, 2015.

[17] A. Washburn, *High-Confidence Learning from Uncertain Data with High Dimensionality.* PhD thesis, The University of Arizona, 2018.

[18] R. Chen and I. C. Paschalidis, "A robust learning approach for regression models based on distributionally robust optimization," *Journal of Machine Learning Research*, vol. 19, no. 13, 2018.

[19] A. Sinha, H. Namkoong, and J. C. Duchi, "Certifying some distributional robustness with principled adversarial training," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, OpenReview.net, 2018.

[20] J. Blanchet, Y. Kang, and K. Murthy, "Robust wasserstein profile inference and applications to machine learning," *Journal of Applied Probability*, vol. 56, no. 3, pp. 830–857, 2019.

[21] R. Chen and I. C. Paschalidis, "Robust grouped variable selection using distributionally robust optimization," *Journal of Optimization Theory and Applications*, vol. 194, no. 3, pp. 1042–1071, 2022.

[22] F. Luo and S. Mehrotra, "Decomposition algorithm for distributionally robust optimization using wasserstein metric with an application to a class of regression models," *European Journal of Operational Research*, vol. 278, no. 1, pp. 20–35, 2019.

[23] J. Blanchet, K. Murthy, and F. Zhang, "Optimal transport-based distributionally robust optimization: Structural properties and iterative schemes," *Mathematics of Operations Research*, vol. 47, no. 2, pp. 1500–1529, 2022.

[24] J. Li, S. Huang, and A. M.-C. So, "A first-order algorithmic framework for distributionally robust logistic regression," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[25] J. Li, C. Chen, and A. M.-C. So, "Fast epigraphical projection-based incremental algorithms for wasserstein distributionally robust support vector machine," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4029–4039, 2020.

[26] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020.

[27] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization.," *Journal of machine learning research*, vol. 13, no. 2, 2012.

[28] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[29] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, *et al.*, "Least angle regression," *Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[30] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[31] S. Rosset and J. Zhu, "Piecewise linear regularized solution paths," *The Annals of Statistics*, pp. 1012–1030, 2007.

[32] R. J. Tibshirani, J. Taylor, *et al.*, "The solution path of the generalized lasso," *The annals of statistics*, vol. 39, no. 3, pp. 1335–1371, 2011.

[33] L. Gunter and J. Zhu, "Efficient computation and model selection for the support vector regression," *Neural Computation*, vol. 19, no. 6, pp. 1633–1655, 2007.

[34] G. Wang, D.-Y. Yeung, and F. H. Lochovsky, "A new solution path algorithm in support vector regression," *IEEE Transactions on Neural Networks*, vol. 19, no. 10, pp. 1753–1767, 2008.

[35] G. Wang, D.-Y. Yeung, and F. H. Lochovsky, "Two-dimensional solution path for support vector regression," in *Proceedings of the 23rd international conference on Machine learning*, pp. 993–1000, 2006.

[36] M. Karasuyama, N. Harada, M. Sugiyama, and I. Takeuchi, "Multi-parametric solution-path algorithm for instance-weighted support vector machines," *Machine learning*, vol. 88, no. 3, pp. 297–330, 2012.

[37] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the support vector machine," *Journal of Machine Learning Research*, vol. 5, no. Oct, pp. 1391–1415, 2004.

[38] C.-J. Ong, S. Shao, and J. Yang, "An improved algorithm for the solution of the regularization path of support vector machine," *IEEE Transactions on Neural Networks*, vol. 21, no. 3, pp. 451–462, 2010.

[39] J. Dai, C. Chang, F. Mai, D. Zhao, and W. Xu, "On the svmpath singularity," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 11, pp. 1736–1748, 2013.

[40] C. G. Sentelle, G. C. Anagnostopoulos, and M. Georgiopoulos, "A simple method for solving the svm regularization path for semidefinite kernels," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 4, pp. 709–722, 2015.

[41] J. Zhu, S. Rosset, R. Tibshirani, and T. J. Hastie, "1-norm support vector machines," in *Advances in neural information processing systems*, p. None, Citeseer, 2003.

[42] B. Gu, J.-D. Wang, G.-S. Zheng, and Y.-C. Yu, "Regularization path for ν-support vector classification," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 5, pp. 800–811, 2012.

[43] B. Gu and V. S. Sheng, "A robust regularization path algorithm for ν-support vector classification," *IEEE Transactions on neural networks and learning systems*, vol. 28, no. 5, pp. 1241–1248, 2016.

[44] L. Wang, J. Zhu, and H. Zou, "The doubly regularized support vector machine," *Statistica Sinica*, pp. 589–615, 2006.

[45] G. Tang and N. Fan, "A survey of solution path algorithms for regression and classification models," *Annals of Data Science*, vol. 9, no. 4, pp. 749–789, 2022.

[46] Y. Shi, *Advances in Big Data Analytics: Theory, Algorithms and Practices*. Springer Nature, 2021.

[47] D. L. Olson, Y. Shi, and Y. Shi, *Introduction to business data mining*, vol. 10. McGraw-Hill/Irwin New York, 2007.

[48] Y. Shi, Y. Tian, G. Kou, Y. Peng, and J. Li, *Optimization based data mining: theory and applications*. Springer Science & Business Media, 2011.

[49] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[50] J. M. Tien, "Internet of things, real-time decision making, and artificial intelligence," *Annals of Data Science*, vol. 4, no. 2, pp. 149–178, 2017.

[51] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[52] K. P. Bennett, J. Hu, X. Ji, G. Kunapuli, and J.-S. Pang, "Model selection via bilevel optimization," in *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pp. 1922–1929, IEEE, 2006.

[53] W. W. Tso, B. Burnak, and E. N. Pistikopoulos, "Hy-pop: Hyperparameter optimization of machine learning models through parametric programming," *Computers & Chemical Engineering*, vol. 139, p. 106902, 2020.

[54] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A comprehensive survey of loss functions in machine learning," *Annals of Data Science*, pp. 1–26, 2020.

[55] L. Wang, M. D. Gordon, and J. Zhu, "Regularized least absolute deviations regression and an efficient algorithm for parameter tuning," in *Sixth International Conference on Data Mining (ICDM'06)*, pp. 690–700, IEEE, 2006.

[56] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[57] R. J. Vanderbei, "Loqo: An interior point code for quadratic programming," *Optimization methods and software*, vol. 11, no. 1-4, pp. 451–484, 1999.

[58] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[59] E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Neural networks for signal processing VII. Proceedings of the 1997 IEEE signal processing society workshop*, pp. 276–285, IEEE, 1997.

[60] V. Vapnik, *The nature of statistical learning theory.* Springer science & business media, 1999.

[61] W. Mao, G. Yan, and L. Dong, "Weighted solution path algorithm of support vector regression based on heuristic weight-setting optimization," *Neurocomputing*, vol. 73, no. 1-3, pp. 495–505, 2009.

[62] W.-t. Mao, L.-l. Dong, and G. Zhang, "Weighted solution path algorithm of support vector regression for abnormal data," in *2008 19th International Conference on Pattern Recognition*, pp. 1–4, IEEE, 2008.

[63] R. Koenker and G. Bassett Jr, "Regression quantiles," *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

[64] I. Takeuchi, Q. V. Le, T. D. Sears, and A. J. Smola, "Nonparametric quantile estimation," *Journal of machine learning research*, vol. 7, no. Jul, pp. 1231–1264, 2006.

[65] Y. Li, Y. Liu, and J. Zhu, "Quantile regression in reproducing kernel hilbert spaces," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 255–268, 2007.

[66] I. Takeuchi, K. Nomura, and T. Kanamori, "Nonparametric conditional density estimation using piecewise-linear solution path of kernel quantile regression," *Neural Computation*, vol. 21, no. 2, pp. 533–559, 2009.

[67] S. Rosset, "Bi-level path following for cross validated solution of kernel quantile regression.," *Journal of Machine Learning Research*, vol. 10, no. 11, 2009.

[68] M. V. McCrea, H. D. Sherali, and A. A. Trani, "A probabilistic framework for weather-based rerouting and delay estimations within an airspace planning model," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 4, pp. 410–431, 2008.

[69] A. Shashua and A. Levin, "Taxonomy of large margin principle algorithms for ordinal regression problems," *Advances in neural information processing systems*, vol. 15, pp. 937–944, 2002.

[70] W. Chu and S. S. Keerthi, "Support vector ordinal regression," *Neural computation*, vol. 19, no. 3, pp. 792–815, 2007.

[71] B. Gu and C. Ling, "A new generalized error path algorithm for model selection," in *International Conference on Machine Learning*, pp. 2549–2558, 2015.

[72] B. Gu, "A regularization path algorithm for support vector ordinal regression," *Neural Networks*, vol. 98, pp. 114–121, 2018.

[73] M. Y. Park and T. Hastie, "L1-regularization path algorithm for generalized linear models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 69, no. 4, pp. 659–677, 2007.

[74] J. K. Tay, B. Narasimhan, and T. Hastie, "Elastic net regularization paths for all generalized linear models," *arXiv preprint arXiv:2103.03475*, 2021.

[75] X. Wang and P. M. Pardalos, "A survey of support vector machines with uncertainties," *Annals of Data Science*, vol. 1, no. 3-4, pp. 293–309, 2014.

[76] E. L. Allgower and K. Georg, "Continuation and path following," *Acta numerica*, vol. 2, pp. 1–64, 1993.

[77] C. Sentelle, G. C. Anagnostopoulos, and M. Georgiopoulos, "Efficient revised simplex method for svm training," *IEEE transactions on neural networks*, vol. 22, no. 10, pp. 1650–1661, 2011.

[78] K. Scheinberg, "An efficient implementation of an active set method for svms," *Journal of Machine Learning Research*, vol. 7, no. Oct, pp. 2237–2257, 2006.

[79] L. Wang, J. Zhu, and H. Zou, "Hybrid huberized support vector machines for microarray classification," in *Proceedings of the 24th international conference on Machine learning*, pp. 983–990, 2007.

[80] L. Wang, J. Zhu, and H. Zou, "Hybrid huberized support vector machines for microarray classification and gene selection," *Bioinformatics*, vol. 24, no. 3, pp. 412–419, 2008.

[81] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural computation*, vol. 12, no. 5, pp. 1207–1245, 2000.

[82] D. Poole, *Linear algebra: A modern introduction*. Nelson Education, 2014.

[83] C.-F. Lin and S.-D. Wang, "Fuzzy support vector machines," *IEEE transactions on neural networks*, vol. 13, no. 2, pp. 464–471, 2002.

[84] Y. Lin, Y. Lee, and G. Wahba, "Support vector machines for classification in nonstandard situations," *Machine learning*, vol. 46, no. 1-3, pp. 191–202, 2002.

[85] X. Yang, Q. Song, and Y. Wang, "A weighted support vector machine for data classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 21, no. 05, pp. 961–976, 2007.

[86] Y. Lee and Z. Cui, "Characterizing the solution path of multicategory support vector machines," *Statistica Sinica*, pp. 391–409, 2006.

[87] Y. Lee, Y. Lin, and G. Wahba, "Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 67–81, 2004.

[88] P. A. Flach, "The geometry of roc space: understanding machine learning metrics through roc isometrics," in *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 194–201, 2003.

[89] M. Majnik and Z. Bosnić, "Roc analysis of classifiers in machine learning: A survey," *Intelligent data analysis*, vol. 17, no. 3, pp. 531–558, 2013.

[90] F. R. Bach, D. Heckerman, and E. Horvitz, "Considering cost asymmetry in learning classifiers," *The Journal of Machine Learning Research*, vol. 7, pp. 1713–1741, 2006.

[91] K.-R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE transactions on neural networks*, vol. 12, no. 2, pp. 181–201, 2001.

[92] G. Wang, D.-Y. Yeung, and F. H. Lochovsky, "A kernel path algorithm for support vector machines," in *Proceedings of the 24th international conference on machine learning*, pp. 951–958, 2007.

[93] D. M. Tax and R. P. Duin, "Support vector domain description," *Pattern recognition letters*, vol. 20, no. 11-13, pp. 1191–1199, 1999.

[94] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[95] G. Lee and C. D. Scott, "The one class support vector machine solution path," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 2, pp. II–521, IEEE, 2007.

[96] M. Karasuyama and I. Takeuchi, "Suboptimal solution path algorithm for support vector machine," *arXiv preprint arXiv:1105.0471*, 2011.

[97] V. S. Sheng and C. X. Ling, "Thresholding for making classifiers cost-sensitive," in *AAAI*, pp. 476–481, 2006.

[98] G. I. Karakoulas and J. Shawe-Taylor, "Optimizing classifers for imbalanced training sets," in *Advances in neural information processing systems*, pp. 253–259, 1999.

[99] B. Schölkopf, A. J. Smola, F. Bach, *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[100] M. A. Davenport, R. G. Baraniuk, and C. D. Scott, "Tuning support vector machines for minimax and neyman-pearson classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1888–1898, 2010.

[101] H. Masnadi-Shirazi and N. Vasconcelos, "Risk minimization, probability elicitation, and cost-sensitive svms.," in *ICML*, pp. 759–766, Citeseer, 2010.

[102] B. Gu, V. S. Sheng, K. Y. Tay, W. Romano, and S. Li, "Cross validation through two-dimensional solution surface for cost-sensitive svm," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1103–1121, 2016.

[103] F. De Morsier, D. Tuia, M. Borgeaud, V. Gass, and J.-P. Thiran, "Semi-supervised novelty detection using svm entire solution path," *IEEE transactions on geoscience and remote sensing*, vol. 51, no. 4, pp. 1939–1950, 2013.

[104] M. Markou and S. Singh, "Novelty detection: a review—part 1: statistical approaches," *Signal processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[105] R. Koenker and K. F. Hallock, "Quantile regression," *Journal of economic perspectives*, vol. 15, no. 4, pp. 143–156, 2001.

[106] A. Christmann and I. Steinwart, "How svms can estimate quantiles and the median," in *Advances in neural information processing systems*, pp. 305–312, 2008.

[107] I. Steinwart, A. Christmann, *et al.*, "Estimating conditional quantiles with the help of the pinball loss," *Bernoulli*, vol. 17, no. 1, pp. 211–225, 2011.

[108] X. Huang, L. Shi, and J. A. Suykens, "Solution path for pin-svm classifiers with positive and negative τ values," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 7, pp. 1584–1593, 2016.

[109] R. Khemchandani, S. Chandra, *et al.*, "Twin support vector machines for pattern classification," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 29, no. 5, pp. 905–910, 2007.

[110] Y. Tian and Z. Qi, "Review on: twin support vector machines," *Annals of Data Science*, vol. 1, no. 2, pp. 253–277, 2014.

[111] Y. Xu, Z. Yang, and X. Pan, "A novel twin support-vector machine with pinball loss," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 2, pp. 359–370, 2016.

[112] Z. Yang, X. Pan, and Y. Xu, "Piecewise linear solution path for pinball twin support vector machine," *Knowledge-Based Systems*, vol. 160, pp. 311–324, 2018.

[113] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 133–142, 2002.

[114] K. Z. Arreola, T. Gärtner, G. Gasso, and S. Canu, "Regularization path for ranking svm," in *ESANN*, 2008.

[115] P. Bloomfield and W. L. Steiger, *Least absolute deviations: theory, applications, and algorithms*. Springer, 1983.

[116] P. J. Huber, "Robust Estimation of a Location Parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73 – 101, 1964.

[117] P. J. Huber, "Robust regression: asymptotics, conjectures and monte carlo," *The annals of statistics*, pp. 799–821, 1973.

[118] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," *Advances in neural information processing systems*, vol. 9, 1996.

[119] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[120] P. M. Esfahani and D. Kuhn, "Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations," *Mathematical Programming*, vol. 171, no. 1, pp. 115–166, 2018.

[121] H. Rahimian and S. Mehrotra, "Distributionally robust optimization: A review," *arXiv preprint arXiv:1908.05659*, 2019.

[122] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[123] T. Hastie, R. Tibshirani, and R. J. Tibshirani, "Extended comparisons of best subset selection, forward stepwise selection, and the lasso," *arXiv preprint arXiv:1707.08692*, 2017.

[124] D. Bertsimas, J. Dunn, C. Pawlowski, and Y. D. Zhuo, "Robust classification," *INFORMS Journal on Optimization*, vol. 1, no. 1, pp. 2–34, 2019.

[125] W. Wiesemann, D. Kuhn, and M. Sim, "Distributionally robust convex optimization," *Operations Research*, vol. 62, no. 6, pp. 1358–1376, 2014.

[126] D. Faccini, F. Maggioni, and F. A. Potra, "Robust and distributionally robust optimization models for linear support vector machine," *Computers & Operations Research*, vol. 147, p. 105930, 2022.

[127] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.