

Nonparametric learning with matrix-valued predictors in high dimensions

Abstract

We consider the problem of learning the relationship between a binary label response and a high-dimensional matrix-valued predictor. Such data problems arise commonly in brain imaging studies, sensor network localization, and personalized medicine. Existing regression analysis often takes a parametric procedure by imposing a pre-specified relationship between variables. However, parametric models are insufficient in capturing complex regression surfaces defined over high-dimensional matrix space. Here, we propose a flexible nonparametric framework for various learning tasks, including classification, level set estimation, and regression, that specifically accounts for the matrix structure in the predictors. Unlike classical approaches, our method adapts to the possibly non-smooth, non-linear pattern in the regression function of interest. The proposal achieves prediction and interpretability simultaneously via a joint optimization of prediction rules and dimension reduction in the matrix space. Generalization bounds, estimation consistency, and convergence rate are established. We demonstrate the advantage of our method over previous approaches through simulations and applications to **XXX** data analyses.

Keywords: Nonparametric learning, matrix-valued predictors, high dimension, classification, level-set estimation, regression.

1 Introduction

2 Methods

Consider a statistical learning problem where we would like to model the relationship between a feature $\mathbf{X} \in \mathcal{X}$ and a response $Y \in \mathcal{Y}$. Suppose that we observe a sample of n data points, $(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)$, identically and independently distributed (i.i.d.) according to a unknown distribution $\mathbb{P}(\mathbf{X}, Y)$ over $\mathcal{X} \times \mathcal{Y}$. We are interested in predicting a new response Y_{n+1} from a new feature value \mathbf{X}_{n+1} . The observations $\{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ are called the training data and $(\mathbf{X}_{n+1}, Y_{n+1})$ the test point. When no confusion arises, we often omit the subscript $(n+1)$ and simply write (\mathbf{X}, Y) for the prototypical test point. The test point is assumed independent of the training data and is drawn from the same unknown distribution \mathbb{P} . Our goal is to make accurate prediction under a wide range of distributions. In particular, we consider a non-parametric, distribution-free setting with no strong assumptions on the data generative distribution other than i.i.d.

We focus on the scenario with matrix-valued predictors and binary label response; that is, $\mathcal{X} = \mathbb{R}^{d_1 \times d_2}$ and $\mathcal{Y} = \{-1, 1\}$. Matrix-valued predictors ubiquitously arise in modern applications. One example is from electroencephalography studies of alcoholism. The data set records voltage value measured from 64 channels of electrodes on 256 subjects for 256 time points (Zhou and Li, 2014). Each feature is a 256×64 matrix and the response is a binary indicator of subject being alcoholic or control. Another example is pedestrian detection from image data. Each image is divided into 9 regions where local orientation statistics are generated with a total of 22 numbers per region. This yields a 22×9 matrix-valued feature and a binary label response indicating whether the image is pedestrian (Shashua et al., 2004).

In the above two examples and many other studies, researchers are interested in *interpretable prediction*, where the goal is to not only make accurate prediction but also identify features that are informative to the prediction. While classical learning algorithms have been successful in prediction with vector-valued predictors, the key challenge with matrix-valued predictors is the complex structure in the feature space. A naive approach is to transform the feature matrices to vectors and apply classical methods based on vectors to solve the problem. However, this vectorization would destroy the structural information of the data matrices. Moreover, the reshaping matrices to vectors results in high dimensionality which leads to overfitting. Notably, the ambient dimension with matrix-valued feature, $d_1 d_2$, is often comparable to, or even larger than the number of sample, n . Our method exploits the structural information in the data matrix to overcome these challenges.

2.0 Three main problems

Before we proceed with our proposal for matrix-valued features, we present the concrete learning problems of our interest. We consider three major supervised learning problems: classification, level set estimation, and regression estimation.

2.2 The problem of classification: Classification is the problem of identifying to which of a set of categories a new observation belongs, based on training samples. We aim to find a decision rule $g(\mathbf{X})$ that has small error

$$\mathbb{P}_{\mathbf{X},y}(y \neq g(\mathbf{X})),$$

where $g(\mathbf{X}) = \text{sign}(f(\mathbf{X}))$ and $f : \mathbb{R}^{d_1 \times d_2} \rightarrow \mathbb{R}$ is a decision function. The classification problem has long been interested. Many attempts have been developed and performed well for example, decision tree, nearest neighbor, neural network and

support vector machine to name a few. However, most of methods have focused on vector valued features. In many classification problems, the input features are naturally represented as matrices or tensors rather than vectors. We want to tackle matrix valued classification preserving the matrix structure.

2.3 *The problem of level set estimation:* The π -level set of p given a fixed $\pi \in [0, 1]$ is the set

$$S(\pi) = \{\mathbf{X} \in \mathbb{R}^{d_1 \times d_2} : p(\mathbf{X}) > \pi\}.$$

Accurate and efficient level set estimation plays an important role in many applications. One example can be found in medical decision making. In Osteosarcoma treatment, the degree of tumor necrosis is used to guide the choice of postoperative chemotherapy (Man et al., 2005). Patients with $\geq 90\%$ necrosis is labeled as 1, which is response variable y . Suppose that \mathbf{X} is a feature matrix collected from the patient such as gene expression levels on each tissue. Knowledge of the regression level set is needed to allow effective postoperative chemotherapy without a biopsy. We consider a nonparametric way to estimate the π -level set of the regression function based on classification problem.

2.4 *The problem of regression estimation:* Regression function calculates expectation of y given a feature matrix \mathbf{X} on the basis of a training set of data. In our setting, the regression $\mathbb{E}(y|\mathbf{X})$ is equivalent to the conditional probability $\mathbb{P}(y = 1|\mathbf{X})$ because the class label y is binary. Knowledge about the class probability itself is of significant interest and can tell us the confidence of the outcome of classification. Traditionally, the regression problem is addressed through distribution assumption like logistic regression or linear discriminant analysis (LDA). In many applications,

however, it is often difficult to justify the assumptions made in logistic regression or satisfy the Gaussian assumption in LDA. These issues become more challenging for matrix features because of high dimensionality. We establish distribution free method for estimating the regression function $p(\mathbf{X})$ based on level set estimation.

The three problems represent common learning tasks with increasing difficulties. Classification problem can be completed from level set $S(\frac{1}{2})$ utilizing Bayes rule. The level set estimation problem becomes trivial when we have all information about regression function. Accordingly, classical approach for the three problems is to find a solution for regression first, and address the other two based on the estimation. This is why the regression problem is also called soft classification. However, our approach finds classification rule first and address the level set estimation and regression problem in order. Through the sequence of solving the problems, we successfully solve the problems without assuming probability distribution.

2.1 Choice of decision function space

The aforementioned three problems are formulated as empirical structure minimization over functions with matrices as inputs. In this section, we introduce function classes we consider for the minimization problem. we propose a family of matrix kernels, which are building blocks for defining functions in matrix space. Kernel methods defined on non-vector objects have recently evolved into a rapidly developing branch of learning on structured data. Informally, a matrix kernel is a distance measure between two matrices with the same size using proper notion of similarity. Unlike vectors, matrix-valued inputs represent two-way relationship across rows and columns at a time. Taking into account of this two-way relationship is essential in the kernel development. Our proposed kernel uses

concept from latent factor models and incorporates the two-way similarities via low rank regularization. We start introducing linear function class in matrix space and generalize nonlinear one based on our proposed matrix kernel.

2.1.1 Linear function class

We propose the linear functions as a decision function class, $f(\mathbf{X}) = \langle \mathbf{B}, \mathbf{X} \rangle$, where $\mathbf{B}, \mathbf{X} \in \mathbb{R}^{d_1 \times d_2}$ and $\langle \mathbf{X}, \mathbf{X}' \rangle = \text{Tr}(\mathbf{X}^T \mathbf{X}')$. The inner product is called Frobenius inner product in the space of matrices and is a generalization of the dot product from vector spaces. We impose low-rankness on the linear predictor to consider the degeneracy of the coefficient matrix \mathbf{B} and leverage the structure information within the predictor \mathbf{X} . Specifically, the coefficient matrix has low-rank r usually much smaller than the matrix size $\min(d_1, d_2)$,

$$\mathbf{B} = \mathbf{C}\mathbf{P}^T \text{ where } \mathbf{C} \in \mathbb{R}^{d_1 \times r}, \mathbf{P} \in \mathbb{R}^{d_2 \times r} \text{ and } r \leq \min(d_1, d_2). \quad (1)$$

The low-rankness makes distinction from classical classification problem in vector spaces and preserves structural information of feature matrices. The low rank constraint on coefficient matrix has been proposed in Support Vector Machine (SVM) classification problem (Pirsiavash et al., 2009; Luo et al., 2015) in matrix spaces. However these papers only focus on application to the classification problem. Our paper is not confined in classification but further apply the linear function class to level set estimation and regression estimations. Furthermore, we extend linear case to nonlinear case based on matrix kernel concept in the next section. There are some prior work on matrix/tensor kernels. Most of these methods, however, are inadapative and do not use label information to guide the kernel construction. We propose a new matrix kernel that solves the limitation.

2.1.2 Nonlinear function class

We generalize classical kernel method in vector spaces to matrix spaces. We briefly summarize kernel method for vector features and introduce new notations and operations which will be used later.

It has been popular and successful to extend linear classifiers in vector space to nonlinear classifiers using kernel method. Classical linear classifier finds linear boundaries in the input vector feature space. By introducing feature mapping which maps input feature space to enlarged dimension space, learning nonlinear classifier becomes possible. In fact, we need not specify the feature mapping at all to obtain an optimal function that minimizes pre-determined loss function. Instead, the learning process only requires knowledge of the kernel function that computes inner products in the transformed enlarged space, thereby avoiding heavy computation. We generalize this kernel approach to the case when input feature is matrix-valued.

Before proposing a new matrix feature mapping and kernel, we introduce notations and operations needed later. Let $\phi_i: \mathbb{R}^{d_i} \rightarrow \mathcal{H}_i$ be feature mappings with a classical kernel defined on vectors $K_i: \mathbb{R}^{d_i} \times \mathbb{R}^{d_i} \rightarrow \mathbb{R}$ for $i = 1, 2$. \mathcal{H}_i denotes enlarged feature space by ϕ_i and a possibly infinite dimensional Hilbert space. Let $\mathcal{H}^{d_1 \times d_2} = \{\mathbf{X}: \mathbf{X} = \llbracket x_{ij} \rrbracket, x_{ij} \in \mathcal{H}\}$ denote the collection of d_1 by d_2 matrices with each entry taking value in a Hilbert space \mathcal{H} . Matrix algebraic operations are carried over from operations on real valued matrices. One can check exact definitions of operations in Supplement.

Now we present matrix the matrix kernel and associated feature mapping. We define matrix valued feature mapping over the d_1 -by- d_2 matrix space.

Definition 1. Let $\phi_1: \mathbb{R}^{d_1} \rightarrow \mathcal{H}_1$ and $\phi_2: \mathbb{R}^{d_2} \rightarrow \mathcal{H}_2$ be classical feature mappings defined

on vector space. Then Φ is matrix feature mappings defined on

$$\Phi(\mathbf{X}): \mathbb{R}^{d_1 \times d_2} \rightarrow (\mathcal{H}_1 \times \mathcal{H}_2)^{d_1 \times d_2} \quad (2)$$

$$\mathbf{X} \mapsto \Phi(\mathbf{X}) = \llbracket \Phi(\mathbf{X})_{ij} \rrbracket \text{ where } \Phi(\mathbf{X})_{ij} \stackrel{\text{def}}{=} (\phi_2(\mathbf{X}_{i:}), \phi_1(\mathbf{X}_{:j})).$$

Notice that the matrix feature mapping considers both row-wise and column-wise enlarged features. From the feature mapping, the linear function $f: \mathbb{R}^{d_1 \times d_2} \rightarrow \mathbb{R}$ with respect to enlarged space $\Phi(\mathbf{X}) \in (\mathcal{H}_1 \times \mathcal{H}_2)^{d_1 \times d_2}$ is defined as,

$$f(\mathbf{X}) \stackrel{\text{def}}{=} \langle \mathbf{B}, \Phi(\mathbf{X}) \rangle, \text{ where } \mathbf{B} = \llbracket (\mathbf{b}_i^{\text{row}}, \mathbf{b}_j^{\text{col}}) \rrbracket \in (\mathcal{H}_1 \times \mathcal{H}_2)^{d_1 \times d_2}, \quad (3)$$

$$\text{with } \mathbf{b}_i^{\text{row}} \in \mathcal{H}_1 \text{ and } \mathbf{b}_j^{\text{col}} \in \mathcal{H}_2 \text{ for all } (i, j) \in [d_1] \times [d_2].$$

Notice we impose this structure on \mathbf{B} for identifiability issue. These matrix valued feature mapping (2) and corresponding linear function (3) are generalization from existing classical kernel method in vector spaces and can be extended naturally to tensor case (see Supplement for the details). We assume that the coefficient \mathbf{B} in (3) admits low rank decomposition as in Section 2.1.1,

$$\mathbf{B} = \mathbf{P}_1 \mathbf{C} \mathbf{P}_2^T, \text{ where } \mathbf{P}_1 \in \mathbb{R}^{d_1 \times r}, \mathbf{P}_2 \in \mathbb{R}^{d_2 \times r} \text{ and } \mathbf{C} = \llbracket (\mathbf{c}_i^{\text{row}}, \mathbf{c}_j^{\text{col}}) \rrbracket \in (\mathcal{H}_1 \times \mathcal{H}_2)^{r \times r}, \quad (4)$$

$$\text{with } \mathbf{c}_i^{\text{row}} \in \mathcal{H}_1 \text{ and } \mathbf{c}_j^{\text{col}} \in \mathcal{H}_2 \text{ for all } i, j \in [r].$$

Again, we have the structured \mathbf{C} for identifiability. When feature mapping ϕ_i is identity for $i = 1, 2$ implying the linear case in Section 2.1.1, we show that considered linear functions (3) with low-rank r defined by (4) are equivalent to the linear functions in Section 2.1.1 with the low-rank r constraint (1). Therefore, our matrix feature mapping is generalization of classical feature mapping on vector spaces and extension to nonlinear case from linear functions on matrix features.

Now we define matrix kernel associated with the matrix feature mapping.

Definition 2. Let $K_i(\cdot, \cdot)$ be classical kernels which can be represented as $K_i(\cdot, \cdot) = \langle \phi_i(\cdot), \phi_i(\cdot) \rangle$ for $i = 1, 2$. Let weight matrices $\mathbf{W}_i = \llbracket w_{jk}^{(i)} \rrbracket \in \mathbb{R}^{d_i \times d_i}$ be rank- r semi-positive definite matrices for $i = 1, 2$. Then $\{\mathbf{W}_i, K_i\}_{i=1,2}$ induce matrix kernel defined by

$$\mathbf{K}: \mathbb{R}^{d_1 \times d_2} \times \mathbb{R}^{d_1 \times d_2} \rightarrow \mathbb{R}$$

$$(\mathbf{X}, \mathbf{X}') \mapsto \mathbf{K}(\mathbf{X}, \mathbf{X}') = \sum_{j,k \in [d_1]} w_{jk}^{(1)} K_1(\mathbf{X}_{j\cdot}, \mathbf{X}'_{k\cdot}) + \sum_{j,k \in [d_2]} w_{jk}^{(2)} K_2(\mathbf{X}_{\cdot j}, \mathbf{X}'_{\cdot k}).$$

The matrix kernel incorporates classical kernel in vector spaces. Like classical kernel, we can associate the feature mapping in Definition 1 with the matrix kernel. Given $\{\mathbf{W}_i, K_i\}_{i=1,2}$, we have

$$\begin{aligned} \mathbf{K}(\mathbf{X}, \mathbf{X}') &= \sum_{j,k \in [d_1]} w_{jk}^{(1)} K_1(\mathbf{X}_{j\cdot}, \mathbf{X}'_{k\cdot}) + \sum_{j,k \in [d_2]} w_{jk}^{(2)} K_2(\mathbf{X}_{\cdot j}, \mathbf{X}'_{\cdot k}) \\ &= \langle (\mathbf{W}_1, \mathbf{W}_2) \Phi(\mathbf{X}), (\mathbf{W}_1, \mathbf{W}_2) \Phi(\mathbf{X}') \rangle. \end{aligned}$$

We can view the matrix kernel as weighted inner product of the feature mappings. From the kernel representation, we learn nonlinear function successfully avoiding specification of feature mapping $\Phi(\mathbf{X})$ as in classical vector case given pre-specified row and column-wise kernels K_1, K_2 .

2.2 Classification

We estimate a decision function $f: \mathbb{R}^{d_1 \times d_2} \rightarrow \mathbb{R}$ over function class introduced in Section 2.1. The decision function class denoted as \mathcal{F} can be either linear or nonlinear. We propose a large margin classifier that minimizes a cost function in f over the function class \mathcal{F} .

$$\min_{f \in \mathcal{F}} n^{-1} \sum_{i=1}^n L(y_i f(\mathbf{X}_i)) + \lambda J(f), \quad (5)$$

where $J(f)$ is a regularization term for model complexity and $L(z)$ is a margin loss that is a function of the functional margin $yf(\mathbf{X})$. Examples of such loss functions are the hinge loss function $L(z) = (1 - z)_+$ and the logistic loss function $L(z) = \log(1 + e^{-z})$. For demonstration, we focus on the hinge loss case in Equation (5). However, our estimation schemes and theorems are applicable to general large-margin classifiers.

We present the solution to (5) with nonlinear kernels which incorporate linear case. Based on the considered decision function class, we solve the following optimization problem,

$$(\hat{\mathbf{P}}_1, \hat{\mathbf{C}}, \hat{\mathbf{P}}_2) = \arg \min_{\{(\mathbf{P}_1, \mathbf{P}_2, \mathbf{C}) \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r} \times (\mathcal{H}_1 \times \mathcal{H}_2)^{r \times r}\}} n^{-1} \sum_{i=1}^n (1 - y_i \langle \mathbf{P}_1 \mathbf{C} \mathbf{P}_2^T, \mathbf{X}_i \rangle)_+ + \lambda \|\mathbf{P}_1 \mathbf{C} \mathbf{P}_2^T\|_F^2. \quad (6)$$

Notice that the optimization problem (6) degenerates to the conventional SVM with vectorized feature matrices when feature mapping is identity and the coefficient is full rank. From the solution to (6), our estimated classifier is written

$$\hat{g}(\mathbf{X}) = \text{sign}(\hat{f}(\mathbf{X})) = \text{sign} \left(\langle \hat{\mathbf{P}}_1 \hat{\mathbf{C}} \hat{\mathbf{P}}_2^T, \mathbf{X} \rangle \right). \quad (7)$$

We make a remark on the implication of the formulation (7). The solution (7) implies a joint learning of dimension reduction and classification risk minimization. This is one of our contribution to combine two different processes into one. To check this, we see the a dual representation of the solution to (6),

$$f(\mathbf{X}) = \sum_{i=1}^n \alpha_i y_i \left(\sum_{j,k \in [d_1]} [\mathbf{H}_{\mathbf{P}_1}]_{jk}, K_1(\mathbf{X}_{j:}^{(i)}, \mathbf{X}_{k:}) + \sum_{j,k \in [d_2]} [\mathbf{H}_{\mathbf{P}_2}]_{jk}, K_2(\mathbf{X}_{:j}^{(i)}, \mathbf{X}_{:k}) \right), \quad (8)$$

where $\{\alpha_i\}_{i=1}^n$ are (sparse) dual solution to (6) and $\mathbf{X}_{jk}^{(i)}$ denotes (j,k)-entry of \mathbf{X}_i . Notice that the representation (8) can be viewed as an element of reproducing kernel Hilbert

space (RKHS) induced by matrix kernel with weight matrices $\{\mathbf{H}_{\mathbf{P}_i}\}_{i=1,2}$ and row and column-wise kernels K_1, K_2 . Therefore, our considered function space \mathcal{F} can be written as

$$\begin{aligned}\mathcal{F} &= \{f: \mathbf{X} \mapsto \langle \mathbf{P}_1 \mathbf{C} \mathbf{P}_2^T, \Phi(\mathbf{X}) \rangle | \mathbf{C} \in (\mathcal{H}_1 \times \mathcal{H}_2)^{d_1 \times d_2} \text{ and } \mathbf{P}_i \in \mathbb{R}^{d_i \times r} \text{ for } i = 1, 2\} \\ &= \{f \in \text{RKHS induced by matrix kernel } \mathbf{K} \text{ with } \{\mathbf{H}_{\mathbf{P}_i}, K_i\}_{i=1,2}\}.\end{aligned}$$

Given row and column-wise kernels, we estimate weight matrices and an element of RKHS induced by the estimated weight matrices. The projection matrices $\{\mathbf{H}_{\mathbf{P}_i}\}_{i=1,2}$ play role in reducing the feature dimension and at the same time, we find the best element of RKHS that minimizes the classification risk by estimating coefficients $\boldsymbol{\alpha}$ in (8). The procedure is summarized as the following optimization

$$\max_{f \in \mathcal{F}} L(f) = \max_{\substack{\text{rank}(\mathbf{W}_i) \leq r, \\ \mathbf{W}_i \geq 0, i=1,2}} \max_{f \in \text{RKHS}(\mathbf{K})} L(f).$$

2.3 Level set estimation

We propose weighted loss function from (5) to estimate the level set,

$$\min_{f \in \mathcal{F}} n^{-1} \sum_{i=1}^n \omega_{\pi}(y_i) L(y_i f(\mathbf{X}_i)) + \lambda J(f), \quad (9)$$

where $\omega_{\pi}(y) = 1 - \pi$ if $y = 1$ and π if $y = -1$. The weighted loss accepts unequal costs for positive and negative misclassifications in margin classifier, where π is the known cost for the negative and $1 - \pi$ is for the positive classes. Notice that equal cost $\pi = \frac{1}{2}$ make (9) reduce to (5). The optimizer to Equation (9) with respect to all measurable function class yields an consistent estimate of the Bayes rule $g_{\pi}(\mathbf{X}) = \text{sign}(f_{\pi}(\mathbf{X}))$ where $f_{\pi}(\mathbf{X}) = p(\mathbf{X}) - \pi$ (Lin et al., 2002; Wang et al., 2008). Therefore, under the considered decision function class, we obtain a minimizer \hat{f}_{π} to (9) and estimate the level set as

$$\hat{S}(\pi) = \{\mathbf{X} \in \mathbb{R}^{d_1 \times d_2} : \text{sign}(\hat{f}_{\pi}(\mathbf{X})) = 1\}. \quad (10)$$

2.4 Regression function estimation

We propose a method to estimate the regression function $p(\mathbf{X}) \stackrel{\text{def}}{=} \mathbb{E}(y = 1|\mathbf{X})$ at any \mathbf{X} which does not necessarily belong to the observed training data set. Non-smooth, non-continuous regression functions are allowed in our framework. Consider the following two steps of approximation to the target function.

$$\begin{aligned} p(\mathbf{X}) &\stackrel{\text{step1}}{\approx} \sum_{h=1}^H \frac{1}{H} \mathbb{1} \left\{ \mathbf{X} : p(\mathbf{X}) \leq \frac{h}{H} \right\} \\ &= \sum_{h=1}^H \frac{1}{H} \mathbb{1} \left\{ \mathbf{X} \notin S \left(\frac{h}{H} \right) \right\} \\ &\stackrel{\text{step2}}{\approx} \sum_{h=1}^H \frac{1}{H} \mathbb{1} \left\{ \mathbf{X} \notin \hat{S} \left(\frac{h}{H} \right) \right\}. \end{aligned}$$

Step 1 approximates the target probability by linear combination of step functions where H is a smooth parameter. In step 2, we plug in the level set estimation defined in (10) given $\pi = h/H$. Here we use consistency of level set estimation. Therefore, we estimate the regression function as,

$$\hat{p}(\mathbf{X}) = \sum_{h=1}^H \frac{1}{H} \mathbb{1} \left\{ \mathbf{X} \notin \hat{S} \left(\frac{h}{H} \right) \right\},$$

by repeatedly estimating the level sets as (10) with different π values, say $\pi = \frac{h}{H}$ for $h = 1, \dots, H$.

3 Algorithm (**concatenated version**)

In this Section, we describe an algorithm to seek the optimizer of Equation (5) in the case of hinge loss function $L(z) = (1 - z)_+$. We consider nonlinear decision function class $\mathcal{F} =$

$\{f: \mathbf{X} \mapsto \langle \mathbf{C}\mathbf{P}^T, \Phi(\mathbf{X}) \rangle | \mathbf{C} = (\mathbf{C}_1, \mathbf{C}_2) \in \mathcal{H}_1^{d_1} \times \mathcal{H}_2^{d_2} \text{ and } \mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2) \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}\}$
given row and columnwise kernels K_1, K_2 . Notice Equation (6) is written as

$$\min_{\substack{\mathbf{C} \in \mathcal{H}_1^{d_1} \times \mathcal{H}_2^{d_2}, \\ \mathbf{P} \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}}} \frac{1}{2} \|\mathbf{C}\mathbf{P}^T\|_F^2 + C \sum_{i=1}^n \xi_i, \quad (11)$$

subject to $y_i \langle \mathbf{C}\mathbf{P}^T, \Phi(\mathbf{X}_i) \rangle \leq 1 - \xi_i$ and $\xi_i \geq 0, i = 1, \dots, n$.

Optimization problem (11) is non-convex problem because low-rank constraint makes feasible set non-convex. We propose to utilize coordinate descent algorithm that solves one block holding the other block fixed. From this approach, we can solve a convex problem in each step. To be specific, first we update \mathbf{C} holding \mathbf{P} fixed. The dual problem of Equation (11) with fixed \mathbf{P} is

$$\max_{\boldsymbol{\alpha}=(\alpha_1, \dots, \alpha_n)} - \sum_{i=1}^n \alpha_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \Phi(\mathbf{X}_i) \mathbf{P} (\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T, \Phi(\mathbf{X}_j) \mathbf{P} (\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T \rangle$$

subject to $0 \leq \alpha_i \leq C, i = 1, \dots, n$.

We use quadratic programming to solve the dual problem and update \mathbf{C} as

$$\mathbf{C} = \sum_{i=1}^n \alpha_i y_i \Phi(\mathbf{X}_i) \mathbf{P} (\mathbf{P}^T \mathbf{P})^{-1} \in \mathcal{H}_r^r \times \mathcal{H}_c^r. \quad (12)$$

We use the formula (12) without information about feature mapping $\Phi(\cdot)$. Second, we assume that \mathbf{C} is fixed and update \mathbf{P} . The dual problem of Equation (11) with fixed \mathbf{C} is

$$\max_{\boldsymbol{\alpha}=(\alpha_1, \dots, \alpha_n)} - \sum_{i=1}^n \alpha_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \mathbf{C} ((\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T \Phi(\mathbf{X}_i)), \mathbf{C} ((\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T \Phi(\mathbf{X}_j)) \rangle, \quad (13)$$

subject to $0 \leq \alpha_i \leq C, i = 1, \dots, n$,

We can find an optimizer of (13) based on kernel information only. We obtain the following formula by plugging (12) into components of (13).

$$\begin{aligned} \mathbf{C}^T \mathbf{C} &= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T \mathbf{K}(i, j) \mathbf{P} (\mathbf{P}^T \mathbf{P})^{-1} \in \mathbb{R}^{r \times r} \times \mathbb{R}^{r \times r}, \\ \mathbf{C}^T \Phi(\mathbf{X}_i) &= \sum_{j=1}^n \alpha_j y_j (\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T \mathbf{K}(i, j) \in \mathbb{R}^{r \times d_1} \times \mathbb{R}^{r \times d_2}, \end{aligned}$$

where $\mathbf{K}(i, j) \stackrel{\text{def}}{=} (\Phi_1(\mathbf{X}_i)^T \Phi_1(\mathbf{X}_j), \Phi_2(\mathbf{X}_i)^T \Phi_2(\mathbf{X}_j)) \in \mathbb{R}^{d_1 \times d_1} \times \mathbb{R}^{d_2 \times d_2}$. Notice that $[\Phi_1(\mathbf{X}_i)^T \Phi_1(\mathbf{X}_j)]_{ss'} = K_1(\mathbf{X}_{s:}^{(i)}, \mathbf{X}_{s':}^{(j)})$ and vice versa for $\Phi_2(\cdot)$. Therefore, we update \mathbf{P} from an optimal coefficient $\boldsymbol{\alpha}$ to (13) without specifying feature mapping.

$$\mathbf{P} = \sum_{i=1}^n \alpha_i y_i (\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T \Phi(\mathbf{X}_i).$$

We end up obtaining nonlinear function output of the form,

$$\begin{aligned} \hat{f}(\mathbf{X}) &= \sum_{k=1}^n \hat{\alpha}_k y_k \left(\sum_{i=1}^{d_1} \sum_{j=1}^{d_1} [\hat{\mathbf{P}}_1 (\hat{\mathbf{P}}_1^T \hat{\mathbf{P}}_1)^{-1} \hat{\mathbf{P}}_1^T]_{ij} K_r([\mathbf{X}_k]_{i:}, [\mathbf{X}]_{j:}) \right. \\ &\quad \left. + \sum_{i=1}^{d_2} \sum_{j=1}^{d_2} [\hat{\mathbf{P}}_2 (\hat{\mathbf{P}}_2^T \hat{\mathbf{P}}_2)^{-1} \hat{\mathbf{P}}_2^T]_{ij} K_c([\mathbf{X}_k]_{:i}, [\mathbf{X}]_{:j}) \right). \end{aligned} \quad (14)$$

Algorithm 1 gives the full description for classification.

By the similar way with little modification, we can obtain an algorithm for weighted margin classifier (9). From the explanation in Section 2.3 and 2.4, we summarize level set and regression estimation procedure in Algorithm 2.

Algorithm 1: Classification algorithm

Input: $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_n, y_n)$, rank r , and pre-specified kernels K_1, K_2

Initilize: $\mathbf{P}^{(0)} \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}$

Do until converges

Update \mathbf{C} fixing \mathbf{P} :

 Solve $\max_{\alpha} - \sum_{i=1}^n \alpha_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \Phi(\mathbf{X}_i), \Phi(\mathbf{X}_j) \mathbf{P}(\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T \rangle$

$$\mathbf{C} = \sum_{i=1}^n \alpha_i y_i \Phi(\mathbf{X}_i) \mathbf{P}(\mathbf{P}^T \mathbf{P})^{-1}.$$

Update \mathbf{P} fixing \mathbf{C} :

 Solve $\max_{\alpha} - \sum_{i=1}^n \alpha_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \Phi(\mathbf{X}_i), \mathbf{C} ((\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T \Phi(\mathbf{X}_j)) \rangle$.

$$\mathbf{P} = \sum_{i=1}^n \alpha_i y_i (\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T \Phi(\mathbf{X}_i).$$

Output: \hat{f} of the form (14)

4 Theory

5 Conclusion

SUPPLEMENTARY MATERIAL

A Hilbert-space valued matrix operations

Proposition 1. *Let $\mathbf{A} = \llbracket a_{ij} \rrbracket$ and $\mathbf{A}' = \llbracket a'_{ij} \rrbracket$ be two matrices in $\mathcal{H}^{d_1 \times d_2}$ and $\mathbf{B} = \llbracket b_{ij} \rrbracket$ be a matrix in $\mathcal{H}^{d_2 \times d_3}$. Let $\mathbf{P} \in \mathbb{R}^{r \times d_1}$ and $\mathbf{P}' \in \mathbb{R}^{d_2 \times r}$ be real valued matrices. Then, we have well defined operations.*

- *Inner product:* $\langle \mathbf{A}, \mathbf{A}' \rangle = \sum_{ij} \langle a_{ij}, a'_{ij} \rangle \in \mathbb{R}$.

Algorithm 2: Level set & Regression Algorithm

Input: $(\mathbf{X}_1, y_1), \dots, (\mathbf{X}_n, y_n)$, rank r , pre-specified kernels K_1, K_2 , and smooth parameter H .

Initialize: $\pi_h = (h - 1)/H$ for $h = 1, \dots, H + 1$

For $h = 1, \dots, H + 1$:

Level set $\hat{S}(\pi_h)$ **estimation:**

Train weighted margin classifier \hat{f}_{π_h} from (9) based on Algorithm 1.

$$\hat{S}(\pi_h) = \{\mathbf{X} \in \mathbb{R}^{d_1 \times d_2} : \text{sign}(\hat{f}_{\pi_h}(\mathbf{X})) = 1\}.$$

Regression $\hat{p}(\mathbf{X})$ **estimation:**

$$\hat{p}(\mathbf{X}) = \sum_{h=1}^H \frac{1}{H} \mathbb{1} \left\{ \mathbf{X} \notin \hat{S}(\pi_h) \right\}.$$

Output: Level sets $\hat{S}(\pi_h)$ for $h = 1, \dots, H$ and regression function $\hat{p}(\mathbf{X})$.

- *Linear combination:* $\mathbf{P}\mathbf{A} = \llbracket c_{ij} \rrbracket \in \mathcal{H}^{r \times d_2}$ where $c_{ij} = \sum_{k \in [d_1]} p_{ik} a_{kj}$ for all $(i, j) \in [r] \times [d_2]$ and $\mathbf{A}\mathbf{P}' = \llbracket c'_{ij} \rrbracket \in \mathcal{H}^{d_1 \times r}$ where $c'_{ij} = \sum_{k \in [d_1]} a_{ik} p'_{kj}$ for all $(i, j) \in [d_1] \times [r]$.
- *Summation:* $\mathbf{A} + \mathbf{A}' = \llbracket a_{ij} + a'_{ij} \rrbracket \in \mathcal{H}^{d_1 \times d_2}$.
- *Matrix product:* $\mathbf{A}\mathbf{B}' = \llbracket c_{ij} \rrbracket \in \mathbb{R}^{d_1 \times d_3}$, where $c_{ij} = \sum_{k \in [d_2]} \langle a_{ik}, b_{kj} \rangle$, for all $(i, j) \in [d_1] \times [d_3]$.

References

Lin, Y., Y. Lee, and G. Wahba (2002). Support vector machines for classification in nonstandard situations. *Machine learning* 46(1-3), 191–202.

- Luo, L., Y. Xie, Z. Zhang, and W.-J. Li (2015). Support matrix machines. In *International conference on machine learning*, pp. 938–947.
- Man, T.-K., M. Chintagumpala, J. Visvanathan, J. Shen, L. Perlaky, J. Hicks, M. Johnson, N. Davino, J. Murray, L. Helman, et al. (2005). Expression profiles of osteosarcoma that can predict response to chemotherapy. *Cancer research* 65(18), 8142–8150.
- Pirsiavash, H., D. Ramanan, and C. C. Fowlkes (2009). Bilinear classifiers for visual recognition. In *Advances in neural information processing systems*, pp. 1482–1490.
- Shashua, A., Y. Gdalyahu, and G. Hayun (2004). Pedestrian detection for driving assistance systems: single-frame classification and system level performance. *IEEE Intelligent Vehicles Symposium, 2004*, 1–6.
- Wang, J., X. Shen, and Y. Liu (2008). Probability estimation for large-margin classifiers. *Biometrika* 95(1), 149–167.
- Zhou, H. and L. Li (2014). Regularized matrix regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76(2), 463–483.