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 arises 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 model is insufficient in capturing complex regression surfaces with respect to high-dimensional matrix-valued predictors. 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 redXXX 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 $\boldsymbol{X} \in \mathcal{X}$ and a response $Y \in \mathcal{Y}$. Suppose that we observe a sample of of n data points, $(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)$, identically and independently distributed (i.i.d.) according to a unknown distribution $\mathbb{P}(\boldsymbol{X}, Y)$ over $\mathcal{X} \times \mathcal{Y}$. We are interested in predicting a new response Y_{n+1} from a new feature value $\boldsymbol{X}_{n+1}$. The observations $\{(\boldsymbol{X}_i, Y_i)\}_{i=1}^n$ are called the training data and $(\boldsymbol{X}_{n+1}, Y_{n+1})$ the test point. When no confusion arises, we often omit the subscript $(n+1)$ and simply write $(\boldsymbol{X}, Y)$ 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 setting 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 (**?**). Each feature data is 256×64 matrix and the response is binary indicator of subject being alcoholic or control. Another example is pedestrian detection from image data. Each image was 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 matrix and a binary label response indicating whether the image is pedestrian (**?**).

In the above two examples and many other studies, researchers are interested in *inter-*

pretable prediction, where the goal is to not only make accurate prediction but also identify important features that are informative to the prediction. A common challenge in the aforementioned two examples is the complex structure in the feature space. Notably, the ambient dimension with matrix-valued feature is $d_1 d_2$, while the number of sample is n. Modern applications are often in the high dimensional regime where $d_1 d_2$ is comparable to, or even larger, than the sample size n. (do we allow d to increase with n in the asymptotic result?). ...(to be continued)

2.0 Three main problems

We consider three major problems on this setting: 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 establish a decision rule $g(\boldsymbol{X})$ that has small error

$$\mathbb{P}_{\boldsymbol{X},y}\left(y \neq g(\boldsymbol{X})\right),$$

where $g(\boldsymbol{X}) = \text{sign}(f(\boldsymbol{X}))$ and $f : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}$ 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. One example is a study of an electroencephalography data set of alcoholism. The data records voltage value measured from 64 channels of electrodes on 256 subjects for 256 time points, so

each feature data is 256×64 matrix and the response is binary indicator of subject being alcoholic or control (**?**). Another example is pedestrian detection from image data. Each image was divided into 9 regions where local orientation statistics were generated with a total of 22 numbers per region, so each feature data is 22×9 matrix and the response is whether the image is pedestrian (**?**). We want to tackles 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

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

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 (**?**). Patients with $\geq 90\%$ necrosis is labeled as 1, which is response variable y. Suppose that $\boldsymbol{X}$ 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 $\boldsymbol{X}$ on the basis of a training set of data. In our setting, the regression $\mathbb{E}(y|\boldsymbol{X})$ is equivalent to the conditional probability $\mathbb{P}(y = 1|\boldsymbol{X})$ 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 $p(\boldsymbol{X})$ based on level set estimation.

The three problems require more and more information sequentially. 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.

The three problems have some challenges and require a new method because the input dataset consists of matrices not vectors. When utilizing classical methods based on vectors to solve the problems, we have to transform the feature matrices to vectors. However, this vectorization would destroy the structural information of the data matrices. Moreover, the reshaping matrices to vectors results in very high dimensionality which leads to overfitting problem. We propose a new methodology for those problems. Our method exploits the structural information of the data matrix. We take advantage of low-rank assumption to describe such structure and overcome overfitting problem.

2.1 Choice of decision function space

Here, we consider a set of linear predictors as decision function class and extend to non-linear case in Section 4. We impose low-rankness on a linear predictor of the form $f(\boldsymbol{X}) = \langle \boldsymbol{B}, \boldsymbol{X} \rangle$, where $\boldsymbol{B}, \boldsymbol{X} \in \mathbb{R}^{d_1 \times d_2}$ and $\langle \boldsymbol{X}, \boldsymbol{X'} \rangle = \mathrm{Tr}(\boldsymbol{X}^T \boldsymbol{X'})$. Specifically, the coefficient matrix $\boldsymbol{B}$ has low-rank r usually much smaller than the matrix size $\min(d_1, d_2)$,

$$\boldsymbol{B} = \boldsymbol{U}\boldsymbol{V}^T \text{ where } \boldsymbol{U} \in \mathbb{R}^{d_1 \times r}, \boldsymbol{V} \in \mathbb{R}^{d_2 \times r} \text{ and } r \leq \min(d_1, d_2).$$

The condition determines trade-off between model complexity and flexibility. This low-rankness makes distinction from classical classification problem for feature vectors and preserves structural information of feature matrices.

2.2 Classification

We consider a large margin classifier that minimizes a cost function in f over a decision function class $\mathcal{F}$

$$\min_{f \in \mathcal{F}} n^{-1} \sum_{i=1}^{n} L\left(y_i f(\boldsymbol{X}_i)\right) + \lambda J(f), \tag{1}$$

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 $yf(\boldsymbol{X})$. 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 (2.2). However, our estimation schemes and theorems are applicable to general large-margin classifiers. Based on the decision function class in Section 2.1, we solve the following optimization problem

$$(\hat{\boldsymbol{U}}, \hat{\boldsymbol{V}}) = \underset{\{(\boldsymbol{U}, \boldsymbol{V}) \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}\}}{\arg\min} n^{-1} \sum_{i=1}^{n} \left(1 - y_i \langle \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{X}_i \rangle\right)_+ + \lambda \|\boldsymbol{U}\boldsymbol{V}^T\|_F^2. \tag{2}$$

Notice that when the coefficient matrix has full rank, the optimization problem (2.2) degenerates to the conventional linear SVM with vectorized feature matrices. From the solution to (2.2), our estimated decision rule is

$$\hat{g}(\boldsymbol{X}) = \text{sign}\left(\langle \hat{\boldsymbol{U}}.\hat{\boldsymbol{V}}^T, \boldsymbol{X}\rangle\right)$$

We make two remarks on the implication of our formulation (2.2). First, the formulation (2.2) 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 (2.2).

$$f(\boldsymbol{X}) = \sum_{i=1}^{n} \alpha_i y_i \langle \boldsymbol{P}_r \boldsymbol{X}_i, \boldsymbol{P}_r \boldsymbol{X}\rangle, \tag{3}$$

where $\{\alpha_i\}_{i=1}^n$ are (sparse) dual solution of (2.2) and $\boldsymbol{P}_r \in \mathbb{R}^{r\times d_1}$ is the projection matrix induced by low rank coefficient $\boldsymbol{U}, \boldsymbol{V}$. The projection matrix plays role in reducing the feature dimension. From the representation, we see that the optimization (2.2) finds the best projection matrix and coefficient that reduce feature dimension and minimize the classification risk at the same time. Second, the dual representation (2.2) can be viewed as an element of the reproducing kernel Hilbert space (RKHS) induced by rank-r linear kernel defined as

$$\langle \boldsymbol{X}, \boldsymbol{X}'\rangle_{\boldsymbol{P}_r} \stackrel{\text{def}}{=} \langle \boldsymbol{P}_r \boldsymbol{X}, \boldsymbol{P}_r \boldsymbol{X}'\rangle, \quad \text{for all } \boldsymbol{X}, \boldsymbol{X}' \in \mathbb{R}^{d_1\times d_2}.$$

From the rank-r linear kernel $\langle\cdot,\cdot\rangle_{\boldsymbol{P}_r}$, the solution (2.2) is written $f(\cdot) = \sum_{i=1}^n \alpha_i y_i \langle \boldsymbol{X}_i, \cdot\rangle_{\boldsymbol{P}_r}$, which consist of RKHS. This RKHS perspective of the solution is a key ingredient to expand to nonlinear case in Section 4.

2.3 Level set estimation

We propose weighted loss function from (2.2) to estimate the level set.

$$\min_{f\in\mathcal{F}} n^{-1}\sum_{i=1}^{n}\omega_\pi(y_i)L\left(y_i f(\boldsymbol{X}_i)\right)+\lambda J(f), \tag{4}$$

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 (2.3) reduce to (2.2). The optimizer to Equation (2.3) with respect to all measurable function class yields an consistent estimate of the Bayes rule $g_\pi(\boldsymbol{X})=\text{sign}\left(f_\pi(\boldsymbol{X})\right)$ with $f_\pi(\boldsymbol{X})=p(\boldsymbol{X})-\pi$ (**??**). Therefore, under the considered decision function class as in Section 2.1, we obtain a minimizer $\hat{f}_\pi$ to (2.3) and estimate the level set as

$$\hat{S}(\pi)=\{\boldsymbol{X}:\mathbb{R}^{d_1\times d_2}:\text{sign}(\hat{f}_\pi(\boldsymbol{X}))=1\}. \tag{5}$$

2.4 Regression function estimation

We propose a method to estimate the regression function $p(\boldsymbol{X})\overset{\text{def}}{=}\mathbb{E}(y=1|\boldsymbol{X})$ at any $\boldsymbol{X}$ which does not necessarily belong to the observed training data set. Linearity in the candidate function space does not rule out nonlinear regression functions. In fact, non-smooth, non-continuous regression functions are allowed in our framework. Consider the

following two steps of approximation to the target function.

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

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 (2.3) in Section 2.3 given $\pi = h/H$. Here we use consistency of level set estimation. Therefore, we estimate the regression function,

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

by repeatedly estimating the level sets in (2.3) with different π values, say $\pi = \frac{h}{H}$ for $h = 1, \ldots, H$.

3 Algorithm

In this Section, we describe the algorithm to seek the optimizer of Equation (2.2) in the case of hinge loss function $L(z) = (1 - z)_+$ and linear function class $\mathcal{F} = \{f : f(\cdot) = \langle \boldsymbol{U}\boldsymbol{V}^T, \cdot \rangle,$ where $\boldsymbol{U} \in \mathbb{R}^{d_1 \times r} \boldsymbol{V} \in \mathbb{R}^{d_2 \times r}\}$. Equation (2.2) is written as

$$\min_{\{(\boldsymbol{U},\boldsymbol{V}) \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}\}} n^{-1} \sum_{i=1}^{n} \left(1 - y_i \langle \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{X}_i \rangle\right)_+ + \lambda \|\boldsymbol{U}\boldsymbol{V}^T\|_F^2$$

We optimize Equation (2.2) with a coordinate descent algorithm that solves one block holding the other block fixed. Each step is a convex optimization and can be solved with

quadratic programming. To be specific, when we fix $\boldsymbol{V}$ and update $\boldsymbol{U}$ we have the following equivalent dual problem

$$\max_{\boldsymbol{\alpha}\in\mathbb{R}^n:\boldsymbol{\alpha}\geq 0}\left(\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\langle \boldsymbol{X}_i,\boldsymbol{X}_j\boldsymbol{V}(\boldsymbol{V}^T\boldsymbol{V})^{-1}\boldsymbol{V}^T\rangle\right)$$
$$\text{subject to}\quad \sum_{i=1}^{N}y_i\alpha_i=0,\quad 0\leq\alpha_i\leq\frac{1}{2\lambda n},\quad i=1.\cdots,n,$$

We use quadratic programming to solve this dual problem and update $\boldsymbol{U}=\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{X}_i\boldsymbol{V}(\boldsymbol{V}^T\boldsymbol{V})^{-1}$. Similar approach is applied to update $\boldsymbol{V}$ fixing $\boldsymbol{U}$. The Algorithm 1 gives the full description.

Algorithm 1: Linear classification algorithm

Input: $(\boldsymbol{X}_1,y_1),\cdots,(\boldsymbol{X}_n,y_m)$, rank r

Parameter: U,V

Initizlize: $\boldsymbol{U}^{(0)},\boldsymbol{V}^{(0)}$

Do until converges

Update $\boldsymbol{U}$ fixing $\boldsymbol{V}$:

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

$\boldsymbol{U}=\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{X}_i\boldsymbol{V}(\boldsymbol{V}^T\boldsymbol{V})^{-1}$.

Update $\boldsymbol{V}$ fixing $\boldsymbol{U}$:

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

$\boldsymbol{V}=\sum_{i=1}^{n}\alpha_i y_i\boldsymbol{X}_i^T\boldsymbol{U}(\boldsymbol{U}^T\boldsymbol{U})^{-1}$.

Output: $\boldsymbol{B}=\boldsymbol{U}\boldsymbol{V}^T$

4 Extension to nonlinear case

We extend linear function class to non-linear class with kernel trick. We enlarge feature space through feature mapping $\boldsymbol{h} : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}^{d_1 \times d_2'}$. Once this mapping fixed, the procedure is the same as before. We fit the linear classifier using pair of input feature and label $\{\boldsymbol{h}(\boldsymbol{X}_i), y_i\}_{i=1}^n$. Define a nonlinear low rank kernel in similar way to linear case.

$$\langle \boldsymbol{X}, \boldsymbol{X}' \rangle_{\boldsymbol{P}_r, h} \stackrel{\text{def}}{=} \langle \boldsymbol{P}_r h(\boldsymbol{X}), \boldsymbol{P}_r h(\boldsymbol{X}') \rangle = \text{trace}\left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}') \boldsymbol{P}_r^T \boldsymbol{P}_r\right] \quad \text{for all } \boldsymbol{X}, \boldsymbol{X}' \in \mathbb{R}^{d_1 \times d_2},$$

where $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}') \stackrel{\text{def}}{=} h(\boldsymbol{X}) h^T(\boldsymbol{X}') \in \mathbb{R}^{d_1 \times d_1}$ denotes the matrix product of mapped features. The solution function $f(\cdot)$ of (2.2) on enlarged feature can be written

$$f(\cdot) = \sum_{i=1}^n \alpha_i y_i \langle \boldsymbol{P}_r h(\boldsymbol{X}_i), \boldsymbol{P}_r h(\cdot) \rangle = \sum_{i=1}^n \alpha_i y_i \langle \boldsymbol{X}_i, \cdot \rangle_{\boldsymbol{P}_r, h} = \sum_{i=1}^n \alpha_i y_i \text{trace}\left[\boldsymbol{K}(\boldsymbol{X}_i, \cdot)\ \boldsymbol{P}_r^T \boldsymbol{P}_r\right],$$

which involves feature mapping $h(\boldsymbol{X})$ only thorough inner products. In fact, we need not specify the the transformation $h(\boldsymbol{X})$ at all but only requires knowledge of the $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}')$. A sufficient condition and a necessary condition for $\boldsymbol{K}$ being reasonable appear in Supplement. Three popular choices for $\boldsymbol{K}$ are

- Linear kernel: $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}') = \boldsymbol{X}\boldsymbol{X}'^T$.
- Polynomial kernel with degree m: $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}') = (\boldsymbol{X}\boldsymbol{X}'^T + \lambda \boldsymbol{I})^{\circ m}$.
- Gaussian kernel: the (i, j)-th entry of $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}')$ is

$$[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}')]_{(i,j)} = \exp\left\{-\frac{1}{2\sigma^2} \|\boldsymbol{X}[i, :] - \boldsymbol{X}'[j, :]\|_2^2\right\}$$

  for all $(i, j) \in [d_1] \times [d_1]$.

One can check detailed description for non-linear case algorithm in Supplement.

5 Theory

6 Conclusion

SUPPLEMENTARY MATERIAL

References

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

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.

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.