Nonparametric learning with matrix-valued predictors in high dimensions

Chanwoo Lee[1], Lexin Li[2], Helen Zhang[3], and Miaoyan Wang[1]

[1]University of Wisconsin-Madison [2]University of California-Berkley [3]University of Arizona

Problems & Existing methods

Problems : Let $\{(\boldsymbol{X}_i, y_i) \in \mathbb{R}^{d_1 \times d_2} \times \{-1, 1\} : i = 1, \ldots, n\}$ denote an i.i.d. sample from unknown distribution $\mathcal{X} \times \mathcal{Y}$.

- Classification: How to efficiently classify high-dimensional matrices with limitied sample size:

$$n \ll d_1 d_2 = \text{dimension of feature space?}$$

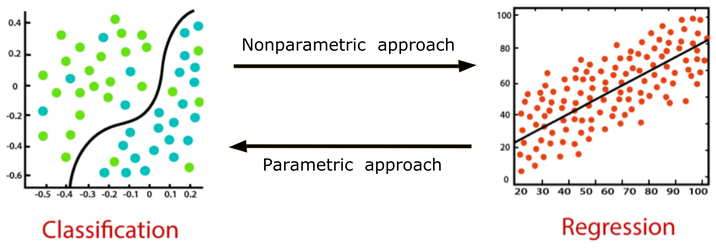
- Regression: How to robustly predict the label probability when little is known to function:

$$\mathbb{P}(y = 1|\boldsymbol{X}) \stackrel{\text{def}}{=} \boldsymbol{p}(\boldsymbol{X})?$$

Existing methods :

- Classification: Decision tree, Nearest neighbor, Neural network, and Support vector machine. However, most of methods have focused on vector valued features.
- Regression: Logistic regression and Linear discriminant analysis. However, it is often difficult to justify the assumptions made when features are matrices because of high-dimensionality.

Goal : We propose nonparametric learning approach with matrix-valued predictors. Unlike classical approach, our approach find classification rule first and address regression problem.

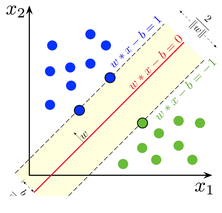


Methods: 1. Classification

- We develop a large-margin classifiers for matrix predictors.

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i f(\boldsymbol{X}_i)) + \lambda J(f), \qquad (1)$$

- We set $\mathcal{F} = \{f : f(\cdot) = \langle \boldsymbol{B}, \cdot \rangle$ where $\text{rank}(\boldsymbol{B}) \le r, \|\boldsymbol{B}\|_F \le C\}$, $J(f) = \|\boldsymbol{B}\|_F^2$, and, $L(x) = (1 - x)_+$.
- We can extend linear classifiers to nonlinear classifiers with a new definition of matrix feature mapping.



Large margin classifier for vector predictors (Picture source: Wiki).

Methods: 2. Regression

- We consider weighted loss function from (1),

$$\hat{f}_\pi = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \omega_\pi(y_i) L(y_i f(\boldsymbol{X}_i)) + \lambda J(f),$$

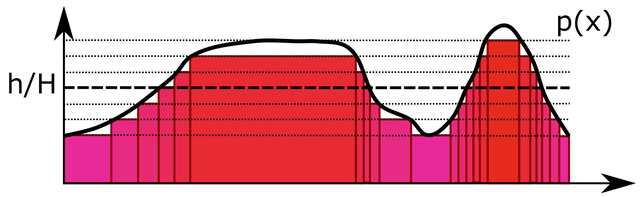
where $\omega_\pi(y) = 1 - \pi$ if $y = 1$ and π if $y = -1$.

- We estimate $\boldsymbol{p}(\boldsymbol{X})$ through two steps of approximations:

$$\boldsymbol{p}(\boldsymbol{X}) \stackrel{\text{1st step}}{\approx} \frac{1}{H} \sum_{h \in [H]} \mathbb{1}\left\{\boldsymbol{X} : \boldsymbol{p}(\boldsymbol{X}) \le \frac{h}{H}\right\}$$
$$\stackrel{\text{2nd step}}{\approx} \frac{1}{H} \sum_{h \in [H]} \mathbb{1}\left\{\boldsymbol{X} : \text{sign}\left[\hat{f}_{\frac{h}{H}}(\boldsymbol{X})\right] = -1\right\},$$

where $H \in \mathbb{N}_+ \to \infty$ is the smoothing parameter.

- 1st step is discretization



- 2nd step is from

$$\mathbb{1}\left\{\boldsymbol{X} : \underbrace{\text{sign}\left[\hat{f}_\pi(\boldsymbol{X})\right] = -1}_{\text{decision region from classification}}\right\} \xrightarrow{\text{in } p} \mathbb{1}\left\{\boldsymbol{X} : \underbrace{\mathbb{P}(Y = 1|\boldsymbol{X}) \le \pi}_{\text{target sublevel set}}\right\},$$

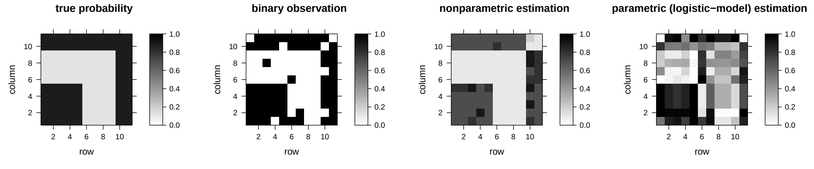
which is verified in [1].

Application: Regression

- Binary matrix probability estimation.
- Goal: estiamte probability matrix $\boldsymbol{P} \in [0, 1]^{d_1 \times d_2}$ from binary observations $\boldsymbol{Y} = \{0, 1\}^{d_1 \times d_2}$ such that $y_{ij} = Bernoulli(p_{ij})$.
- Procedure: 1. Generate training set: $\{(\boldsymbol{X}_{ij}, y_{ij}) : (i, j) \in [d_1] \times [d_2]\}$ where

$$[\boldsymbol{X}_{ij}]_{pq} = \begin{cases} 1 & \text{if } (p, q) = (i, j) \\ 0 & \text{otherwise} \end{cases}.$$

2. Estimate $\hat{\boldsymbol{P}}_{ij} = \mathbb{P}(y_{ij} = 1|\boldsymbol{X}_{ij})$.



Algorithms

- We factor the coefficient matrix $\boldsymbol{B} = \boldsymbol{U}\boldsymbol{V}^T$ where $\boldsymbol{U} \in \mathbb{R}^{d_1 \times r}$ and $\boldsymbol{V} \in \mathbb{R}^{d_2 \times r}$.
- We take alternating optimization approach to solve non-convex problem. (1).

Algorithm 1: Classification algorithm

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

Initizlize: $(\boldsymbol{U}^{(0)}, \boldsymbol{V}^{(0)}) \in \mathbb{R}^{d_1 \times r} \times \mathbb{R}^{d_2 \times r}$

Do until converges

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

$\boldsymbol{U} = \arg\min_{\boldsymbol{U}} \frac{1}{n} \sum_{i=1}^{n} \left(1 - \langle \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{X}_i \rangle\right)_+ + \lambda \|\boldsymbol{U}\boldsymbol{V}^T\|_F.$

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

$\boldsymbol{V} = \arg\min_{\boldsymbol{V}} \frac{1}{n} \sum_{i=1}^{n} \left(1 - \langle \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{X}_i \rangle\right)_+ + \lambda \|\boldsymbol{U}\boldsymbol{V}^T\|_F.$

Output: $\hat{f}(\boldsymbol{X}) = \langle \boldsymbol{U}\boldsymbol{V}^T, \boldsymbol{X} \rangle$

Theoretical results

Theorem 1. *Assume that $\{\boldsymbol{X}_i\}_{i=1}^n$ be set of i.i.d. Gaussian distribution with bounded variation. Then with high probability,*

$$\mathbb{P}[Y \ne \textit{sign}(f^*(\boldsymbol{X}))] - \mathbb{P}[Y \ne \textit{sign}(\hat{f}(\boldsymbol{X}))] \le \frac{4C\sqrt{r(d_1 + d_2)}}{\sqrt{n}},$$

where f^* is the best predictor in $\mathcal{F}$.

Theorem 2. *Denote $\hat{\boldsymbol{p}}$ as an estimated probability function from our method. Under some assumptions, we have*

$$\mathbb{E}\|\hat{\boldsymbol{p}} - \boldsymbol{p}\|_1 = \mathcal{O}\left(\left(\frac{\log\left(n/r(d_1 + d_2)\right)}{(n/r(d_1 + d_2))}\right)^{1/(2-\alpha \wedge 1)}\right),$$

where α is a constant determined by true probability. If $\alpha > 1$ and $d_1 = d_2 = d$, we have

$$\mathbb{E}\|\hat{\boldsymbol{p}} - \boldsymbol{p}\|_1 = \mathcal{O}\left(\frac{\log\left(n/rd\right)}{(n/rd)}\right).$$

References

[1] Junhui Wang, Xiaotong Shen, and Yufeng Liu. “Probability estimation for large-margin classifiers”. In: *Biometrika* 95.1 (2008), pp. 149–167.