Large margin based regression for matrix feature

Abstract

The text of your abstract. 200 or fewer words.

Keywords: 3 to 6 keywords, that do not appear in the title

(tentative)
We consider the problem of learning the relationship between binary outcomes and high-dimensional matrix-valued predictors.
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 relation form 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 accuracy and interpretability by a joint optimization of prediction 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.

General comments:
1. Distinguish model assumption, population statements vs. sample statements, objective function, estimates. Allocate them separately so easier for readers to find.
2. Subjective opinions need to be back up with facts.
3. Math symbols should be defined before the first use.

1 Introduction

blue: math issue.

yellow: logic issue

2 Methods

2.0 Three main problems

Assume that we are given a set of training data $\{(\boldsymbol{X}_1, y_1), \ldots, (\boldsymbol{X}_n, y_n)\}$ with data matrix $\boldsymbol{X}_i \in \mathbb{R}^{d_1 \times d_2}$ and class labels $y_i \in \{+1, -1\}$ drawn according to some unknown probability density function $p_{\boldsymbol{X},y}(\boldsymbol{X}, y)$. Define $p(\boldsymbol{X}) = \mathbb{E}(y|\boldsymbol{X})$. We consider three major problems on this setting: classification, level set estimation, and regression estimation.

i.i.d. is a crucial modeling assumption

p = (E+1)/2

2.1 *The problem of classification*: We aim to address learning a model with a low error on the training data. One successful approach is large-margin classifier. We consider a large margin classifier that minimizes a cost function in f over a decision function class $\mathcal{F}$

be specific.

move to 2.1

1. this is a solution not a problem. 2. restrict to population model/ assumptions in this section. Do not mix with sample quantities.

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

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 $L(z) = \log(1 + e^{-z})$.

Text

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

Do not introduce notation that you never use

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

this is the optimal rule for problem 2.2.
In parallel, you should introduce the optimal rule for problem 2.1

What is the criteria for a good solution?
Add: (1) population risk/objective (2) the optimal decision rule.

Add: connections among three problems.
E.g. which problem is easy, which is hard? connection between 2.1 and 2.2?
classical plug-in approach: 2.3 -> 2.2->2.1. Our approach. 2.1->2.2>2.3. benefit?

We consider the problem of estimating the π-level set of the regression function without assuming any certain models but utilizing the large-margin classifier.

distribution-free, not really model-free.
Any statements/assumptions we made are ``models", e.g. i.i.d, binary, function class, low-rank, etc.

2.3 *The problem of regression estimation*: Based on level set estimation problem, we aim to establish a model-free method for estimating $p(\boldsymbol{X}) = \mathbb{E}(y|\boldsymbol{X})$. 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. This problem is also known as soft classification, since the estimated probability can be used to determine the classification boundary.

When utilizing classical methods based on feature vectors to solve the above 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.

topic sentence?

We propose a methodology for those problems. Our method exploits the structural information of the data matrix. We take advantage of low-rank assumption to describe the correlation within a matrix and overcome overfitting problem.

``correlation" is a probabilistic description. did we ever introduce such structure within X?

2.1. Choice of function space
move ``low-rank assumption" to here.

2.1 Classification

It is well known that the hinge loss enjoys sparseness and robustness, which are two desirable properties for a good classifiers. For demonstration, we focus on the hinge loss case in Equation (1). However, our estimation schemes and theorems are applicable to general large-margin classifiers. We propose a linear predictor with low-rank coefficient matrix of the form $f_{\boldsymbol{B}}(\boldsymbol{X}) = \langle \boldsymbol{B}, \boldsymbol{X} \rangle$, where $\boldsymbol{B}, \boldsymbol{X} \in \mathbb{R}^{d_1 \times d_2}$ and $\langle \boldsymbol{X}, \boldsymbol{X}' \rangle = \mathrm{Tr}(\boldsymbol{X}^T \boldsymbol{X}')$. We can extend the linear model to non-linear case in Section 4. We impose low-rank structure on

Assumption

(Add) We make two remarks on the implications of our formulation (2).
First, the formulation (2) implies a joint learning of dimension reduction and classification risk minimization. This is one of our contribution ….
To see this, ….
Second, the solution to (2) has a dual representation….which is sparse….

the coefficient matrix $\boldsymbol{B}$,

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

Why? the conclusion is fair, but I cannot see how it is simply drawn from the earlier math.

We see that the condition provides a reasonable trade-off between model complexity and model flexibility. This low-rankness makes distinction from classical classification problem for feature vectors and considers structural information of feature matrices. Based on the considered function class, we present the following formulation from Equation (1):

(\hat U,\hat V) = \arg\min

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

Notice that when the rank of the coefficient matrix is full rank, the optimization problem (2) degenerates to the conventional linear SVM with vectorized feature matrices.

add the estimated decision rule here.

(add topic sentence; see top) We show that the solution of (2) has the form (one can check Supplement for the details)

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

where $\{\alpha_i\}_{i=1}^n$ are solution (spare) of the dual problem in (2) and $\boldsymbol{P}_r \in \mathbb{R}^{r \times d_1}$ is the projection matrix induced by low rank coefficient $\boldsymbol{U}, \boldsymbol{V}$. Let $\langle \cdot, \cdot \rangle_{\boldsymbol{P}_r}$ denote the low rank linear kernel for a pair of matrices:

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

The solution function $f(\cdot)$ can be written $f(\cdot) = \sum_{i=1}^n \alpha_i y_i \langle \boldsymbol{X}_i, \cdot \rangle_{\boldsymbol{P}_r}$, which can be viewed as an element in the reproducing kernel Hilbert space induced by rank-r linear kernels $\{\langle \cdot, \cdot \rangle_{\boldsymbol{P}} : \boldsymbol{P} \in \mathbb{R}^{r \times d_1}$ is a projection matrix$\}$. Therefore, we estimate the optimal hyperplane that separate the training data the best on projected feature matrices. The classification rule $\hat{G}(\boldsymbol{X})$ can be written as

why? I did not see the logical flow.

in which sense?

$$\hat{G}(\boldsymbol{X}) = \text{sign}\left(\sum_{i=1}^{n} \hat{\alpha}_i y_i \langle \boldsymbol{X}_i, \boldsymbol{X} \rangle_{\hat{\boldsymbol{P}}_r}\right),$$

What is G? Introduce the estimand in 2.1

where $\{\hat{\alpha}_i\}_{i=1}^n$ are estimated coefficients and $\hat{\boldsymbol{P}}_r$ is an estimated projection matrix. The detailed algorithm for the estimation will appear in Section 3.

2.2 Level set estimation

topic sentence?

Since the large-margin classifier showed good performance in classification, one might expect to extract any information about level set $S(p, \pi)$ from the large-margin classifier. Lin et al. (2002) showed that the solution $\hat{f}$ to (1) targets directly at sign $\left(p(\boldsymbol{X}) - \frac{1}{2}\right)$. Moreover, Wang et al. (2008) proved that the minimizer $\hat{f}_\pi$ to π-weighted loss function is a consistent estimate of $\text{sign}(p(\boldsymbol{X}) - \pi)$. Therefore, we solve the regularization problem with weighed loss function from (1),

not introduced yet.

Not sure whether they still hold under restricted function space. You should add some modifiers "the optimizer to … over all measurable functions is …"

$$\min_{f \in \mathcal{F}} n^{-1} \sum_{i=1}^{n} \omega_\pi(y_i) L\left(y_i f(\boldsymbol{X}_i)\right) + \lambda J(f), \tag{3}$$

not defined yet

where $\omega_\pi(y) = 1 - \pi$ if $y = 1$ and π if $y = -1$. From the solution $\hat{f}_\pi$ to (3), we estimate the level set $\hat{S}(p, \pi) = \{\boldsymbol{X} : \mathbb{R}^{d_1 \times d_2} : \text{sign}(\hat{f}_\pi(\boldsymbol{X})) = 1\}$.

use displayed mode. You will need this equation in section 2.3.

why the argument p?

2.3 Regression function estimation

Our proposed method is designed to estimate the regression function $p(\boldsymbol{X}) \stackrel{\text{def}}{=} \mathbb{E}(y = 1 | \boldsymbol{X})$ at any $\boldsymbol{X}$ which does not necessarily belong to the observed training data set. Consider the following two steps of approximation to the target function.

$$p(\boldsymbol{X}) \overset{\text{step1}}{\approx} \sum_{h=1}^{H} \frac{1}{H} \mathbb{1}\left\{\boldsymbol{X} : p(\boldsymbol{X}) \leq \frac{h}{H}\right\}$$
$$\overset{\text{step2}}{\approx} \sum_{h=1}^{H} \frac{1}{H} \mathbb{1}\left\{\boldsymbol{X} \in \hat{S}^c(p, h/H)\right\},$$

add a paragraph: 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. add a figure.

where $\hat{S}(p,\pi) = \{\boldsymbol{X} : \mathbb{R}^{d_1 \times d_2} : \text{sign}(\hat{f}_\pi(\boldsymbol{X})) = 1\}$. Step 1 approximates the target probability by linear combination of step functions and Step 2 uses the fact that estimated level set in Section 2.2 is consistent. The probability estimation scheme can be summarized as follows.

Shceme

S.1 Choose a sequence of weight $\pi_h = \frac{h}{H}$, for $h = 1, \ldots, H$.

S.2 For each weight $\pi_h \in [0,1]$, solve Equation (3) with $\omega_{\pi_h}(y)$

rewrite.
Leverage the results in section 2.2

S.3 Denote the sequence of solutions and estimated level sets

$$\{\hat{f}_h\}_{h=1}^H \text{ and } \{\hat{S}(p, h/H)\}_{h=1}^H.$$

S.4 Estimate the target probability function by

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

3 Algorithm

In this Section, we describe the algorithm to seek the optimizer of Equation (1) 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 (1) 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) 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

not introduced.

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

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq \frac{1}{2\lambda n}, \quad i = 1. \cdots, n,$$

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 (1) on enlarged feature can be written

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

where both alpha and P are unknown.

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.

Wang, J., X. Shen, and Y. Liu (2008). Probability estimation for large-margin classifiers. *Biometrika 95*(1), 149–167.