Statistics and Its Interface Volume 17 (2024) 231–247

Density-convoluted tensor support vector machines

Boxiang Wang, Le Zhou, Jian Yang, and Qing Mai*

With the emergence of tensor data (also known as multidimensional arrays) in many modern applications such as image processing and digital marketing, tensor classification is gaining increasing attention. Although there is a rich toolbox of classification methods for vector-based data, these traditional methods may not be adequate for tensor data classification. In this paper, we propose a new classifier called density-convoluted tensor support vector machine (DCT-SVM). This method is motivated by applying a kernel density convolution method on the SVM loss to induce a new family of classification loss functions. To establish the theoretical foundation of DCT-SVM, the probabilistic order of magnitude for its excess risk is systematically studied. For efficiently computing DCT-SVM, we develop a fast monotone accelerated proximal gradient descent algorithm and show the convergence of the algorithm. With simulation studies, we demonstrate the superior performance of DCT-SVM over many popular classification methods. We further demonstrate the real potential of DCT-SVM using a modern data application for online advertising.

AMS 2000 subject classifications: Primary 62H30; secondary 62-07.

Keywords and phrases: Kernel density estimation, Large-margin classification, Non-convex optimization, Support vector machines, Tensor data classification.

1. INTRODUCTION

Tensor data are increasingly common in many application areas such as digital marketing, econometrics, finance, image processing, social network analysis, etc. Among these applications, classification on tensor predictors is a ubiquitous task. For example, it is crucial for online advertising companies to identify intended audience to raise their revenues. For a group of ad audience, by summarizing their view counts of p_1 ad campaigns that are delivered on p_2 devices (e.g., phones, tablets, personal computers) and p_3 age groups, we construct a $p_1 \times p_2 \times p_3$ tensor-valued predictor, which can be used to predict whether the proportion of the ads getting clicked after being displayed, namely, the click-through rate, is above some pre-specified level. In particular, with binary labels $y_i \in \{\pm 1\}$ and M-way tensor-valued predictor $\mathcal{X}_i \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_M}$ in the training data $\{y_i, \mathcal{X}_i\}_{i=1}^n$, the goal is to fit a classifier $\hat{f}$ and predict the class label y_{new} for unseen test tensor data $\mathcal{X}_{\text{new}}$ according to the sign of $\hat{f}(\mathcal{X}_{\text{new}})$.

Powerful classification algorithms with high prediction accuracy are paramount to the success of these applications with tensor data. Most classical machine learning algorithm have been developed for vector-valued predictors [15]. Examples of popular algorithms include linear discriminant analysis, logistic regression, naïve Bayes classifiers, kernel density classifiers, neural networks, ensemble learning methods including random forest and gradient boosting, as well as the support vector machines [SVM, 8, 37, 38]. Among these methods, the SVM enjoys nice geometric interpretation without imposing specific model assumptions: it performs classification by directly maximizing the margin between the different classes. The SVM has been shown to be one of the best classifiers in terms of the prediction accuracy through extensive numerical studies on hundreds of benchmark data sets [11].

In contrast to the rich classification toolbox for the vector-valued data, how to classify tensor data is still an active research area. A plausible strategy of classifying tensor data is to unfold all the tensors into vectors and then apply vector-based classifiers; however, vectorization inevitably destructs the intrinsic spatial tensor structure and may yield poor classification accuracy. Hence, in the literature it is a general agreement that tensor data should be analyzed in its original form, and existing vector methods should be coupled with appropriate assumptions to exploit the tensor structure for more accurate classification.

Given the highly competitive classification accuracy of SVM on vector-valued data, it is an appealing and important topic to extend SVM to tensor data analysis. A special challenge of generalizing SVM is efficient computation. Unlike many regression methods which possess closed-form solutions or efficient algorithms, even the vector-valued SVM is computationally challenging: it is either solved on its dual space via the quadratic programming algorithm or formulated as a non-smooth unconstrained optimization problem handled by the subgradient descent. Existing tensor SVM methods often adopt the alternating algorithm to break the tensor SVM into a series of sub-problems that computes the

*Corresponding author.

vector-valued SVM [34, 13]. With these intensive SVM algorithms cycled in every iteration, the alternating algorithm drives the tensor SVM computationally prohibitive. In the meantime, theoretical justifications for existing tensor SVM methods are absent. It is unclear whether these methods produce consistent results in theory.

In this work, to advance both theory and computation of tensor SVM, we formulate a tensor SVM model which handles tensor-valued predictors and also allows some vector-valued predictors in the model to adjust for the prediction. To leverage the tensor structure, we assume that the tensor coefficient satisfies the low-rank assumption that the tensor of interest can be approximated by a low-rank decomposition, namely, the CANDECOMP/PARAFAC (CP) decomposition [17, 19, 5, 31, 49]. The CP decomposition is attractive because it drastically reduces the number of parameters to allow parsimonious modeling. The CP decomposition is also very interpretable and is regarded as a generalizations of singular value decomposition on matrices (two-way tensors). In the literature, [51] initiated a tensor regression model based on the CP decomposition, and they further extended the model to generalized linear models including logistic regression for classification analysis. On the other hand, the low-rank assumption has also be employed by tensor generalizations of linear discriminant analysis [22, 50].

Moreover, to tackle the burden of computing tensor SVM, we employ a technique called kernel density convolution to smooth the SVM loss. This technique has been used in the optimization community, for example, [3] and [30], and was recently used for smoothing quantile regression [10, 16, 33] and the high-dimensional SVM [42]. The use of kernel density convolution on tensor SVM essentially gives birth to a brand-new classifier, which we coin the name Density-Convoluted Tensor SVM (DCT-SVM). Although DCT-SVM is motivated from a computational strategy, in contrast to the use of existing smoothing techniques, we appraise DCT-SVM as a new classifier rather than treating it as an approximated solution of the original SVM. DCT-SVM is indexed by δ, which originates from the bandwidth of kernel density estimation. Rather than restricting $\delta \to 0$ to aim for the SVM loss, we treat the index δ as a tunable hyperparameter of DCT-SVM. We systematically study the learning theory of DCT-SVM under general random design setting. Our theoretical result is the first in the literature that establishes the probabilistic order of magnitude for the excess risk under the tensor large-margin classification framework. Our theory is non-trivial and some new applications of empirical process theory are employed. Regarding the computation, we develop an efficient monotone accelerate proximal gradient descent algorithm to compute DCT-SVM and present the convergence analysis, so we avoid using alternating algorithm which iterates intensive computations.

With the success of tensor SVM, we further extend it to a unified family of large-margin classifiers, which include logistic regression and Huberized SVM as special cases. With extensive numerical studies, we show that the DCT-SVM outperforms the other large-margin classifiers and popular vector-valued classifiers such as boosting, random forest, kernel logistic regression, kernel SVM, and neural nets, as well as a tensor classification method called CATCH which is based on linear discriminant analysis [29]. Last, we apply the method on a real-world application on online advertising to demonstrate the adequate interpretability and predictive power of DCT-SVM. The promising performance of DCT-SVM also corroborates the argument in Chapter 3 of [15] that for prediction purposes linear models can sometimes outperform fancier nonlinear models.

We would like to remark that there are other approaches to exploit the tensor structure, which we do not explore in this paper. For example, [45] fit the logistic regression model with the addition of a variety of tensor norm penalties. [24, 48, 46] consider another type of tensor decomposition called Tucker decomposition [35] to conduct the regression analysis on tensor data. [29, 27] assume that the covariance matrix has a separable structure under the linear discriminant analysis model, while [44] incorporate the envelope model [7] into tensor discriminant analysis. For reviews of recent advances in tensor modeling, we refer interested readers to two survey papers, [4] and [32].

1.1 Notations and structure

We first introduce standard tensor notation and operations [19, for example] that are used frequently in this paper. For positive integers $M \geq 2$, $p_1, \ldots, p_M$, a multi-dimensional array $\mathcal{B} \in \mathbb{R}^{p_1 \times \cdots \times p_M}$ is referred to as an *M-way tensor*. The *vectorization* of a tensor $\mathcal{B}$, $\text{vec}(\mathcal{B})$, is a $(\prod_m p_m \times 1)$ column vector, with $\mathcal{B}_{i_1,\ldots,i_M}$ being its j-th element, where $j = 1 + \sum_{m=1}^{M}(i_m - 1)\prod_{m'=1}^{m-1} p_{m'}$. The *mode-$m$ matricization*, $\mathcal{B}_{(m)}$, is a matrix of dimension $p_m \times \prod_{m' \neq m} p_{m'}$, with $\mathcal{X}_{i_1,\ldots,i_M}$ being its (i_m, j)-th element, where $j = 1 + \sum_{m' \neq m}(i_{m'} - 1)\prod_{l < m', l \neq m} p_l$. If we fix every index of the tensor but one, then we have a *fiber*. For example, $\mathcal{B}_{i_1,\ldots,i_{m-1},I_m,i_{m+1},\ldots,i_M}$ for $I_m \in \{1, \ldots, p_m\}$ forms a $(p_m \times 1)$ vector that is called the mode-k fiber of $\mathcal{B}$.

The mode-m product of a tensor $\mathcal{B}$ and a matrix $\boldsymbol{\alpha} \in \mathbb{R}^{d \times p_m}$, denoted by $\mathcal{B} \times_m \boldsymbol{\alpha}$, is an M-way tensor of dimension $p_1 \times \cdots \times p_{m-1} \times d \times p_{m+1} \times \cdots \times p_M$, with each element being the product of a mode-m fiber of $\mathcal{B}$ and a row vector of $\boldsymbol{\alpha}$. The *mode-m vector product* of a tensor $\mathcal{B}$ and a row vector of $\mathbf{c} \in \mathbb{R}^{p_m}$, denoted by $\mathcal{B} \bar{\times}_m \mathbf{c}$, is an $(M-1)$-way tensor of dimension $p_1 \times \ldots \times p_{m-1} \times p_{m+1} \times \ldots \times p_M$, with each element being the inner product of a mode-m fiber of $\mathcal{B}$ and $\mathbf{c}$. The *inner product* of two tensors of the same dimensions is defined to be $\langle \mathcal{B}, \mathcal{X} \rangle = \text{vec}(\mathcal{B})^\top \text{vec}(\mathcal{X})$. The *outer product* of M vectors $\boldsymbol{\beta}_1 \in \mathbb{R}^{p_1}, \ldots, \boldsymbol{\beta}_M \in \mathbb{R}^{p_M}$ is denoted as $\boldsymbol{\beta}_1 \circ \ldots \circ \boldsymbol{\beta}_M$, which is a $p_1 \times \ldots \times p_M$ tensor whose $(j_1, \ldots, j_M)$-th element is $\prod_{m=1}^{M} \beta_{mj_m}$.

For $a, b \in \mathbb{R}$, let $a \wedge b = \min\{a, b\}$ and $a \vee b = \max\{a, b\}$. For a sequence $\{a_n\}$ and another nonnegative sequence $\{b_n\}$, we write $a_n = O(b_n)$ if there exists a constant $c > 0$ such that $|a_n| \le cb_n$ for all $n \ge 1$. And we write $a_n \asymp b_n$ if $a_n = O(b_n)$ and $b_n = O(a_n)$. Also, we use $a_n = o(b_n)$, or $a_n \ll b_n$, to represent $\lim_{n\to\infty} a_n/b_n = 0$. We write $b_n \gg a_n$ if $a_n \ll b_n$. For nonnegative sequences $\{a_n\}$ and $\{b_n\}$, we also write $a_n \lesssim b_n$ (or $b_n \gtrsim a_n$) if there exists a constant $c > 0$ such that $a_n \le cb_n$ for all $n \ge 1$. Let $\psi : [0,\infty) \to [0,\infty]$ be a nondecreasing, convex function with $\psi(0) = 0$, then we denote $\|Z\|_\psi = \inf\{t > 0 : \mathbb{E}[\psi(|Z|/t)] \le 1\}$ as the ψ-Orlicz norm for any random variable Z. In particular, if $p \ge 1$, let $\psi_p(x) := \exp\{x^p\} - 1$ which is a nondecreasing convex function with $\psi_p(0) = 0$, then we denote its corresponding Orlicz norm as $\|Z\|_{\psi_p} = \inf\{t > 0 : \mathbb{E}[\exp\{|Z|^p/t^p\}] \le 2\}$ where Z is any random variable. For a sequence of random variables $\{Z_n\}_{n\ge1}$, we write $Z_n = O_p(1)$ if $\lim_{M\to\infty}\limsup_{n\to\infty} \mathrm{P}(|Z_n| > M) = 0$, and we write $Z_n = o_p(1)$ if $\lim_{n\to\infty} \mathrm{P}(|Z_n| > \epsilon) = 0, \forall\epsilon > 0$. For two sequences of random variables Z_n and Z'_n, we write $Z_n = O_p(Z'_n)$ if $Z_n/Z'_n = O_p(1)$, and we write $Z_n = o_p(Z'_n)$ if $Z_n/Z'_n = o_p(1)$.

The rest of the paper is organized as follows. In Section 2, we first review SVM for vector-valued data, introduce the tensor SVM with CP decomposition, and propose DCT-SVM. The learning theory and computation algorithm of DCT-SVM are studied in Section 3 and 4, respectively. Section 5 presents numerical example and Section 6 studies the real online advertising application.

2. METHODOLOGY

2.1 Review of support vector machines for vector-valued data

This work focuses on binary classification. Suppose the training sample consists of n data points, $\{y_i, \mathbf{z}_i\}$, where $\mathbf{z}_i$ is a p-dimensional predictor and $y_i \in \{-1, 1\}$ for $i = 1, 2, \dots, n$ is the binary class label. The goal of classification is to find a decision boundary $\{z : f(\mathbf{z}) = 0\}$ so that the label of an unseen data point $\mathbf{z}_0$ is predicted as $\mathrm{sgn}(f(\mathbf{z}_0))$. The class label is incorrectly predicted if $\mathbb{I}_{yf(\mathbf{z})<0}$, which is called the 0-1 loss and $\mathbb{I}$ is the indicator function. The risk of a classifier f is called the 0-1 risk and is defined as $\mathcal{R}(f) = \mathbb{E}\mathbb{I}_{yf(\mathbf{z})<0}$, where the expectation is taken over the data generating distribution. The lowest risk $\mathcal{R}^\star = \min_f \mathbb{E}\mathcal{R}(f)$ is called the Bayes risk and is given by the Bayes classifier $f^\star(\mathbf{z}) = \mathbb{I}_{\eta(\mathbf{z})>1/2}$, where $\eta(\mathbf{z})$ is the conditional probability $P(y = 1|\mathbf{z})$. The term yf is dubbed the *margin.*

Due to the intractable nature of the 0-1 loss, a family of large-margin classifiers are developed to minimize the ϕ-risk, $\mathcal{R}_\phi(f) = \mathbb{E}\phi(yf)$, where $\phi(yf)$ is a convex surrogate of the 0-1 loss and a function of the margin. Among the family of large-margin classifiers, the SVM is widely used in practice. The SVM loss is $\phi_{\text{svm}}(yf) = (1 - yf)_+ = \max\{1 - yf, 0\}$, which is called the hinge loss.

In this work, we focus on linear classifiers, which often offer interpretable descriptions of how predictions are made. Linear classifiers take the form of $\alpha_0 + \mathbf{z}^\top\boldsymbol{\beta}$. When solving the classifier from the training data, the SVM is formulated as

(2.1)
$$(\hat\alpha_0, \hat{\boldsymbol{\beta}}) = \underset{\alpha_0\in\mathbb{R},\ \boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \left[\frac{1}{n}\sum_{i=1}^n \phi_{\text{svm}}\left(y_i(\alpha_0 + \mathbf{z}_i^\top\boldsymbol{\beta})\right) + \lambda\|\boldsymbol{\beta}\|_2^2\right],$$

where $\|\boldsymbol{\beta}\|_2^2 = \boldsymbol{\beta}^\top\boldsymbol{\beta}$ is the ϕ_2 penalty, λ is the shrinkage parameter to be tuned. According to the solution of problem (2.1), the label of an unseen data point $(y_0, \mathbf{z}_0)$ is predicted as $\mathrm{sgn}(\hat\alpha_0 + \mathbf{z}_0^\top\hat{\boldsymbol{\beta}})$.

A family of large-margin classifiers is formed by replacing the hinge loss in problem (2.1) with other large-margin loss functions. Two examples are:

- logistic regression:
$$\phi_{\text{logit}}(yf) = \log(1 + e^{-yf});$$
- Huberized SVM:
$$\phi_{\text{HSVM}}(yf) = \begin{cases} 1 - yf - \delta/2, & yf \le 1-\delta, \\ \dfrac{(1-yf)^2}{2\delta}, & 1-\delta < yf \le 1, \\ 0, & yf > 1, \end{cases}$$

which approximates the SVM hinge loss as $\delta \to 0$.

2.2 Tensor support vector machines with CP decomposition

For tensor data analysis, suppose each data point in the training data $(y_i, \mathbf{z}_i, \mathcal{X}_i)_{i=1}^n$ has a binary label $y_i \in \{-1, 1\}$, vector-valued predictors $\mathbf{z}_i \in \mathbb{R}^{p_0}$, and M-way tensor-valued predictors $\mathcal{X}_i \in \mathbb{R}^{p_1\times p_2\times\cdots\times p_M}$. We consider the general data generating scheme where the training data $(y_i, \mathbf{z}_i, \mathcal{X}_i)_{i=1}^n$ are i.i.d. copies of some random element $(y, \mathbf{z}, \mathcal{X})$, and all the random variables are defined on some common probability space $(\Omega, \mathcal{F}, \mathrm{P})$ with $\mathbb{E}$ being the corresponding expectation.

We consider the linear classifiers that take the form $f(\mathbf{z}, \mathcal{X}) = \alpha_0 + \mathbf{z}^\top\boldsymbol{\alpha} + \langle\mathcal{B}, \mathcal{X}\rangle$, where α_0 is an intercept, $\boldsymbol{\alpha}$ is a vector-valued coefficient, and $\mathcal{B}$ is a tensor-valued coefficient that has the same dimension with the tensor predictor $\mathcal{X}$. To reduce the number of parameters in the classifier $f(\mathbf{z}, \mathcal{X})$, we impose CANDECOMP/PARAFAC (CP) decomposition [17, 19] on the tensor coefficient $\mathcal{B}$:

(2.2)
$$\mathcal{B} = \sum_{r=1}^R \boldsymbol{\beta}^{(1r)} \circ \boldsymbol{\beta}^{(2r)} \cdots \circ \boldsymbol{\beta}^{(Mr)},$$

where $\boldsymbol{\beta}^{(mr)} \in \mathbb{R}^{p_m}$ for each m and r, and R is the rank of the CP decomposition. Therefore the number of parameters in $\mathcal{B}$ is effectively reduced from $\prod_{m=1}^M p_m$ to $R\sum_{m=1}^M p_m$. For ease of exposition, we denote the CP decomposition in equation (2.2) by

$$\mathcal{B} := [\![\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_M]\!],$$

where $\mathbf{B}_m = (\boldsymbol{\beta}^{(m1)}, \boldsymbol{\beta}^{(m2)}, \ldots, \boldsymbol{\beta}^{(mR)}) \in \mathbb{R}^{p_m \times R}$ for each $m = 1, 2, \ldots, M$. Also, the vectors $\boldsymbol{\beta}^{(mr)}$ are not identifiable by themselves. To handle the identifiability issue, we restrict the coefficient in the following convex set,

$$\begin{aligned} S_{\mathcal{B}} = \{[\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!] | \beta_1^{(mr)} = 1,& \\ m = 1, \ldots, M-1,& \\ r = 1, \ldots, R,& \\ \beta_1^{(M1)} \geq \beta_1^{(M2)} \geq \ldots \geq \beta_1^{(MR)} \},& \end{aligned}$$

which is the conventional way of handling the scaling and permutation indeterminacy in the literature; see [51] for example.

With the CP decomposition, we formulate the tensor SVM as

$$\left(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}\right) = \underset{\substack{\alpha_0 \in \mathbb{R}, \boldsymbol{\alpha} \in \mathbb{R}^{p_0} \\ \mathcal{B} = [\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!] \in S_{\mathcal{B}}}}{\operatorname{argmin}} \left[\mathcal{L}(\alpha, \mathcal{B}) + \mathcal{P}(\alpha, \mathcal{B})\right], \tag{2.3}$$

where

$$\begin{aligned} &\mathcal{L}(\alpha, \mathcal{B}) \\ =&\frac{1}{n} \sum_{i=1}^{n} \left\{1 - y_i \left(\alpha_0 + \mathbf{z}_i^\top \boldsymbol{\alpha} \langle [\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!], \mathcal{X}_i \rangle\right)\right\}_+, \end{aligned}$$

and

$$\mathcal{P}(\alpha, \mathcal{B}) = \lambda \left(\|\boldsymbol{\alpha}\|_2^2 + \sum_{m=1}^{M} \|\mathbf{B}_m\|^2\right) \tag{2.4}$$

is the ℓ_2 penalty on $\boldsymbol{\alpha}$ and each $\mathbf{B}_m$. Prediction on an unseen data point $(y_{\text{new}}, \mathbf{z}_{\text{new}}, \mathcal{X}_{\text{new}})$ is thus made according to $\operatorname{sgn}(\hat{\alpha}_0 + \mathbf{z}_{\text{new}}^\top \hat{\boldsymbol{\alpha}} + \langle \hat{\mathcal{B}}, \mathcal{X}_{\text{new}} \rangle)$.

2.3 Smoothing SVM through density convolution

The main computational difficulty for solving problem (2.3) lies in its non-smooth objective function, which results from the non-smoothness of the SVM hinge loss. To handle the computational burden, we employ a kernel density convolution technique to smooth the objective.

We first relate the SVM problem with the distribution of the margin. Given parameters $\alpha_0, \boldsymbol{\alpha}, \mathcal{B}$, we treat the margin

$$T(y, \mathbf{z}, \mathcal{X}) \coloneqq y \left(\alpha_0 + \mathbf{z}^\top \boldsymbol{\alpha} + \langle [\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!], \mathcal{X} \rangle\right)$$

as a new random variable and define $F(t)$ to be its cumulative distribution function (cdf). Hence, the objective function of the tensor SVM can be written as $\int_{-\infty}^{\infty} (1-t)_+ \mathrm{d}F(t)$. Likewise, the unpenalized empirical version of the SVM, i.e., problem (2.3), can be understood as $\int_{-\infty}^{\infty} (1-t)_+ \mathrm{d}\hat{F}(t)$, where the empirical cdf $\hat{F}(t) = n^{-1} \sum_{i=1}^{n} \mathbb{I}_{T(y_i, \mathbf{z}_i, \mathcal{X}_i) \leq t}$ is employed to estimate the cdf of the margin.

To smooth the objective, we use the kernel density estimator in place of the empirical cdf,

$$\tilde{F}(t) = \int_{-\infty}^{t} \frac{1}{n\delta} \sum_{i=1}^{n} K\left(\frac{u - T(y_i, \mathbf{z}_i, \mathcal{X}_i)}{\delta}\right) \mathrm{d}u,$$

thus the objective function of the unpenalized empirical version of the SVM becomes

$$\begin{aligned} &\int_{-\infty}^{\infty} (1-t)_+ d\tilde{F}(t) \\ =&\frac{1}{n} \sum_{i=1}^{n} \int_{-\infty}^{\infty} (1-t)_+ \frac{1}{\delta} K\left(\frac{t - T(y_i, \mathbf{z}_i, \mathcal{X}_i)}{\delta}\right) \mathrm{d}t \\ \coloneqq&\frac{1}{n} \sum_{i=1}^{n} \phi_\delta(T(y_i, \mathbf{z}_i, \mathcal{X}_i)), \end{aligned}$$

inducing a family of new large-margin loss functions, which we call density-convoluted tensor SVM loss (DCT-SVM).

In this work, we consider the Gaussian kernel $K(u) = \frac{1}{\sqrt{2\pi}} \exp\{-u^2/2\}$, which gives the loss function of DCT-SVM with Gaussian kernel:

$$\begin{aligned} \phi_\delta^{\text{Gauss}}(yf) = (1 - yf)\Phi\left(\frac{1 - yf}{\delta}\right)& \\ + \frac{\delta}{\sqrt{2\pi}} \exp\left\{-\frac{(1 - yf)^2}{2\delta^2}\right\},& \end{aligned}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.

We also use the Epanechnikov kernel, $K(t) = 0.75(1 - t^2)\mathbb{I}_{\{|t| \leq 1\}}$, which is commonly used in kernel density estimation and the DCT-SVM loss is:

$$\begin{aligned} &\phi_\delta^{\text{Epan}}(yf) \\ =&\begin{cases} 1 - yf, & yf \leq 1 - \delta, \\ \dfrac{(1 - yf + \delta)^3 (3\delta - (1 - yf))}{16\delta^3}, & 1 - \delta < yf \leq 1 + \delta, \\ 0, & yf \geq 1 + \delta. \end{cases} \end{aligned}$$

Figure 1 plots the DCT-SVM loss functions with $\delta = 1$ using Gaussian and Epanechnikov kernels, SVM hinge loss, Huberized SVM loss with $\delta = 1$, and logistic regression loss. With DCT-SVM loss functions, the induced DCT-SVM classifier is formulated as

$$\begin{aligned} &\left(\hat{\alpha}_{0\delta}, \hat{\boldsymbol{\alpha}}_\delta, \hat{\mathcal{B}}_\delta\right) \\ =& \underset{\substack{\alpha_0 \in \mathbb{R}, \boldsymbol{\alpha} \in \mathbb{R}^{p_0} \\ \mathcal{B} = [\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!] \in S_{\mathcal{B}}}}{\operatorname{argmin}} \Bigg[\frac{1}{n} \sum_{i=1}^{n} \phi_\delta \Big(y_i \Big(\alpha_0 + \mathbf{z}_i^\top \boldsymbol{\alpha} \\ &+ \langle [\![\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_M]\!], \mathcal{X}_i \rangle\Big)\Big) + \mathcal{P}(\alpha, \mathcal{B})\Bigg], \end{aligned} \tag{2.5}$$

where the penalty term was given in equation (2.4).

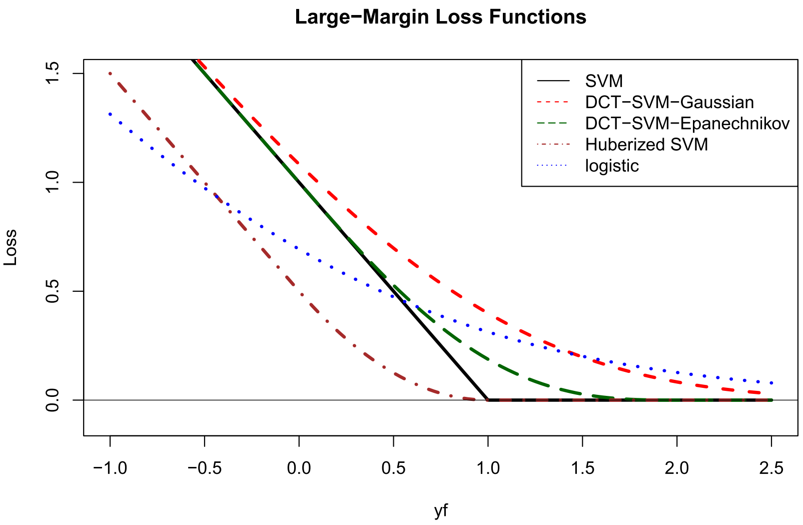

Figure 1. Plots of large-margin loss functions, including SVM, DCT-SVM with Gaussian kernel and $\delta = 1$, DCT-SVM with Epanechnikov kernel and $\delta = 1$, Huberized SVM with $\delta = 1$, and logistic regression, against the margin variable yf.

3. LEARNING THEORY

In this section, we provide the statistical properties of DCT-SVM for tensor data. We impose the following conditions for the choice of kernel function in density convolution.

Assumption 1. *Suppose that the kernel function $K : \mathbb{R} \to [0, \infty)$ satisfies the following properties:*

1. $K(-t) = K(t)$, $\forall t \in \mathbb{R}$;
2. $\inf_{t\in[-r,r]} K(t) > 0$ *for some large enough* $r > 0$;
3. $\int_{-\infty}^{\infty} K(t)\,\mathrm{d}t = 1$.

Assumption 1 is standard and can be easily satisfied by many kernel functions including Gaussian kernels and Epanechnikov kernels.

In this section, our theory holds for the classifier DCT-SVM indexed by any given positive δ; for notation simplicity, we ignore the subscript δ in the loss function and estimators. To establish the theoretical properties of our estimators, we assume $\{(y_i, \mathbf{z}_i, \mathcal{X}_i)\}_{i=1}^n$, $(y, \mathbf{z}, \mathcal{X})$ are independent and identically distributed on $\{-1, 1\} \times \mathbb{R}^{p_0} \times \mathbb{R}^{p_1\times p_2\times\cdots\times p_M}$, and there exists some constant $\Pi > 0$ such that $\max\{\|\mathbf{z}\|, \|\mathrm{vec}(\mathcal{X})\|\} \le \Pi$ almost surely. Also, we search for a solution in a compact parameter space $\{(\alpha_0, \boldsymbol{\alpha}, \mathcal{B}) : \max\{\|\alpha_0\|, \|\boldsymbol{\alpha}\|, \|\mathrm{vec}(\mathcal{B})\|\} \le \Pi'\}$ where $\Pi' > 0$ is some constant. Similar setup is widely considered for nonconvex optimization problems, see for instance, [16, 41].

We define the excess risk associated with parameter value $(\alpha_0, \boldsymbol{\alpha}, \mathcal{B})$,

$$\begin{aligned}\mathcal{R}(\alpha_0, \boldsymbol{\alpha}, \mathcal{B}) :=& \mathbb{E}[\phi\{y(\alpha_0 + \mathbf{z}^\top\boldsymbol{\alpha} + \langle\mathcal{B}, \mathcal{X}\rangle)\}] \\ &- \mathbb{E}[\phi\{y(\alpha_0^* + \mathbf{z}^\top\boldsymbol{\alpha}^* + \langle\mathcal{B}^*, \mathcal{X}\rangle)\}],\end{aligned}$$

where

$$(\alpha_0^*, \boldsymbol{\alpha}^*, \mathcal{B}^*) = \underset{(\alpha_0,\boldsymbol{\alpha},\mathcal{B})\in\Theta}{\operatorname{argmin}}\ \mathbb{E}\left[\phi\left\{1 - y\left(\alpha_0 + \mathbf{z}^\top\boldsymbol{\alpha} + \langle\mathcal{B}, \mathcal{X}\rangle\right)\right\}\right],$$

and $\Theta = \{(\alpha_0, \boldsymbol{\alpha}, \mathcal{B}) : \mathcal{B} \in S_{\mathcal{B}}\}$.

With Assumption 1, the following theorem gives an bound for the excess risk of our estimator $(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}})$.

Theorem 1. *Suppose the kernel function K satisfies the properties in Assumption 1. Choosing $\lambda \asymp n^{-\frac{1}{2}}$, we have*

$$\mathcal{R}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}) = O_p(n^{-\frac{1}{4}}).$$

The above probabilistic order of magnitude for the excess risk also implies the order of magnitude of the risk under the 0-1 loss, which is the following corollary.

Theorem 2. *Suppose the linear function $\bar{\alpha}_0 + \mathbf{z}^\top\bar{\boldsymbol{\alpha}} + \langle\bar{\mathcal{B}}, \mathcal{X}\rangle$ is the solution to $\operatorname{argmin}_f \mathbb{E}[\mathbb{I}_{\{y\neq\mathrm{sign}(f(\alpha_0,\boldsymbol{\alpha},\mathcal{B}))\}}]$ and $\bar{\mathcal{B}} \in S_{\mathcal{B}}$. Let*

$$\begin{aligned}\mathcal{R}_{0-1}(\alpha_0, \boldsymbol{\alpha}, \mathcal{B}) =& \mathbb{E}[\mathbb{I}_{\{y\neq\mathrm{sign}(\alpha_0+\mathbf{z}^\top\boldsymbol{\alpha}+\langle\mathcal{B},\mathcal{X}\rangle)\}}] \\ &- \mathbb{E}[\mathbb{I}_{\{y\neq\mathrm{sign}(\bar{\alpha}_0+\mathbf{z}^\top\bar{\boldsymbol{\alpha}}+\langle\bar{\mathcal{B}},\mathcal{X}\rangle)\}}].\end{aligned}$$

Then, under the conditions of Theorem 1,

$$\mathcal{R}_{0-1}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}) = O_p(n^{-\frac{1}{8}}).$$

Theorems 1 & 2 both indicate that our estimator is consistent, as both risks converge to zero in probability as $n \to \infty$. Such results provide theoretical support for our method. Since we do not impose specific condition on p_i, our theory holds true under both high-dimensional and low-dimensional settings. In particular, in high dimensions, we do not require sparse signal, which is different from the traditional high-dimensional literature where the signal is typically assumed to be sparse.

Our theory focuses on the probabilistic risk bound for tensor classification with ℓ_2 penalty. This is different from works that study the estimation error of the coefficients [50, 24, 44] under the logistic regression or discriminant analysis model. In practice, SVM is usually used to produce an accurate prediction, while researchers rarely try to interpret the coefficients. Hence, our study of the risk bound is more relevant to the application of SVM. On the other hand, literature on high dimensional tensor regression and classification typically adopt ℓ_1-penalty or non-convex penalty and they typically require sparse signal [29, 14]. To the best of our knowledge, our work is the first in literature to study tensor classification with ℓ_2-penalty, without requiring any signal to be sparse. Moreover, the explicit convergence rates appearing in Theorem 1 and Theorem 2 are the first in the literature for tensor SVM.

4. COMPUTATION

In this section, we develop an accelerated proximal gradient descent algorithm to solve problem (2.5). For the sake of presentation, we define a new vector of length $R\sum_{m=1}^{M} p_m + p_0 + 1$:

$$\boldsymbol{\gamma} = \left(\alpha_0, \boldsymbol{\alpha}^\top, \mathrm{vec}(\mathbf{B}_1)^\top, \mathrm{vec}(\mathbf{B}_2)^\top, \ldots, \mathrm{vec}(\mathbf{B}_M)^\top\right)^\top \tag{4.1}$$

assembling all the parameters to be estimated. We then write the objective function of problem (2.5) as $C(\boldsymbol{\gamma})$ and name its feasible set as $S(\boldsymbol{\gamma})$.

We first initialize the algorithm. To accommodate the equality constraint in $S_{\mathcal{B}}$, we fix the first row of each $\mathbf{B}_m^{(0)}$, $m = 1, 2, \ldots, M-1$ to be one, and we initialize all the other elements in $\alpha_0^{(t)}$, $\boldsymbol{\alpha}^{(t)}$, $\mathbf{B}_1^{(0)}, \mathbf{B}_2^{(0)}, \ldots, \mathbf{B}_M^{(0)}$, e.g., from the standard Gaussian distribution. Since the inequality constraint in $S_{\mathcal{B}}$ does not affect the objective value after the equality constraint is satisfied, the inequality constraint is handled at the last step of the algorithm. On the basis of $\alpha_0^{(0)}$, $\boldsymbol{\alpha}^{(0)}$, $\mathbf{B}_1^{(0)}, \mathbf{B}_2^{(0)}, \ldots, \mathbf{B}_M^{(0)}$, $\boldsymbol{\gamma}^{(0)}$ is assembled according to (4.1).

For $t = 0, 1, 2, \ldots$, the proximal gradient descent algorithm updates

$$\boldsymbol{\gamma}^{(t+1)} = \boldsymbol{\gamma}^{(t)} - d_t \nabla C(\boldsymbol{\gamma}^{(t)}),$$

where d_t is the step size to be specified later. To give $\nabla C(\boldsymbol{\gamma}^{(t)})$, we first retrieve each $\alpha_0^{(t)}$, $\boldsymbol{\alpha}^{(t)}$, and each $\mathbf{B}_m^{(t)}$ from $\boldsymbol{\gamma}^{(t)}$ according to (4.1). For each $m = 1, 2, \ldots, M$, let $(\mathcal{X}_i)_{(m)}$ be the mode-m matricization of $\mathcal{X}_i$, define

$$\kappa_i^{(t)} = \alpha_0^{(t)} + \mathbf{z}_i^\top \boldsymbol{\alpha}^{(t)} + \left\langle [\![\mathbf{B}_1^{(t)}, \mathbf{B}_2^{(t)}, \ldots, \mathbf{B}_M^{(t)}]\!], \mathcal{X}_i \right\rangle,$$

construct a $p_m \times R$ matrix $\mathbf{D}_m^{(t)}$ by

$$\begin{aligned}\mathbf{D}_m^{(t)} =& \frac{1}{n}\sum_{i=1}^{n} y_i \phi_\delta' \left\{ y_i \kappa_i^{(t)} \right\} (\mathcal{X}_i)_{(m)} \\ & \left(\mathbf{B}_M^{(t)} \circ \ldots \circ \mathbf{B}_{m+1}^{(t)} \circ \mathbf{B}_{m-1}^{(t)} \circ \ldots \circ \mathbf{B}_1^{(t)}\right) + 2\lambda \mathbf{B}_m^{(t)},\end{aligned}$$

and further zero out the first row of $\mathbf{D}_m^{(t)}$ for each $m = 1, 2, \ldots, M-1$ to ensure the linear constraint in $S_{\mathcal{B}}$ continues to be satisfied by the new solution. Thus, the gradient $\nabla C(\boldsymbol{\gamma}^{(t)}) \in \mathbb{R}^{R\sum p_m + p_0 + 1}$ is given by

$$\begin{aligned}&\left(\frac{1}{n}\sum_{i=1}^{n} y_i \phi_\delta' \left\{ y_i \kappa_i^{(t)} \right\}, \frac{1}{n}\sum_{i=1}^{n} y_i \phi_\delta' \left\{ y_i \kappa_i^{(t)} \right\} \mathbf{z}_i^\top + 2\lambda (\boldsymbol{\alpha}^{(t)})^\top, \right. \\ & \left. \mathrm{vec}(\mathbf{D}_1^{(t)})^\top, \mathrm{vec}(\mathbf{D}_2^{(t)})^\top, \ldots, \mathrm{vec}(\mathbf{D}_M^{(t)})^\top \right).\end{aligned}$$

The step size d_t is determined according to the Barzilai-Borwein rule [2] and backtracking; specifically, let $d_t = 0.5^l a_t$ where

$$a_t = \frac{\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^{(t-1)}\|_{\mathrm{F}}^2}{\langle \boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^{(t-1)}, \nabla C(\boldsymbol{\gamma}^{(t)}) - \nabla C(\boldsymbol{\gamma}^{(t-1)}) \rangle}$$

with l being the smallest integer such that

$$\begin{aligned}C(\boldsymbol{\gamma}^{(t+1)}) \le & C(\boldsymbol{\gamma}^{(t)}) + \left\langle \nabla C(\boldsymbol{\gamma}^{(t)}), \boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^{(t)} \right\rangle \\ & + \frac{1}{2d_t}\left\| \boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^{(t)} \right\|_2^2.\end{aligned}$$

After the algorithm converges, we obtain the solution $\hat{\boldsymbol{\gamma}}$ and retrieve the corresponding units $\hat{\alpha}_0$, $\hat{\boldsymbol{\alpha}}$, $\hat{\mathbf{B}}_1, \hat{\mathbf{B}}_2, \ldots, \hat{\mathbf{B}}_M$. We further swap the columns of $\hat{\mathbf{B}}_M$ to satisfy the inequality constraint in $S_{\mathcal{B}}$. To be specific, find a permutation $\tau : \{1, 2, \ldots, R\} \to \{1, 2, \ldots, R\}$ such that the elements in the first row of $\hat{\mathbf{B}}_M$ follow $\hat{\beta}_1^{M\tau(1)} \ge \hat{\beta}_1^{M\tau(2)} \ge \ldots \ge \hat{\beta}_1^{M\tau(R)}$. We thus use τ to permute all the matrices $\hat{\mathbf{B}}_1, \hat{\mathbf{B}}_2, \ldots, \hat{\mathbf{B}}_M$ so the objective value is unchanged.

The proximal gradient descent algorithm can be further accelerated. Since problem (2.5) is nonconvex, we apply the monotone accelerated proximal gradient (MAPG) algorithm, which is developed in [21] as a special version of Nesterov's acceleration [28]. In addition, we also employ a warm-start strategy to solve problem (2.5) with a decreasing sequence of tuning parameters: $\lambda_1 > \lambda_2 > \ldots > \lambda_L > 0$. For each $l > 1$, the solution obtained for each λ_{l-1} is employed as the initial value for λ_l. We use five initial random starts for λ_1 to alleviate the issue of local minima. For space concern, we do not give the derivation of the MAPG algorithm while we present detailed pseudo-code in Algorithm 1.

The convergence analysis is given in Lemma 1, which follows from Theorem 1 of [21]. The proof is omitted in this work.

Algorithm 1 Monotone Accelerated Proximal Gradient Descent for DCT-SVM

```
Input: $\mathbf{y} \in \{-1,1\}^n$, $\mathbf{z} \in \mathbb{R}^{n\times p}$, $\mathcal{X} \in \mathbb{R}^{n\times p_1 \times p_2 \times \ldots \times p_M}$, $\delta > 0$, and $\lambda_1 > \lambda_2 > \ldots > \lambda_L > 0$.
Output: $\hat{\alpha}_0^{[l]}$, $\hat{\boldsymbol{\alpha}}^{[l]}$, $\hat{\mathbf{B}}_1^{[l]}, \hat{\mathbf{B}}_2^{[l]}, \ldots, \hat{\mathbf{B}}_M^{[l]}$ for each $l$.
 1: Get the density-convoluted SVM loss $\phi_\delta(t)$ in (2.3) and the first-order derivative $\phi'_\delta(t)$.
 2: for $l = 1, \ldots, L$ do
 3:   if $l > 1$ then
 4:     Initialize $\alpha_0^{(0)} = \hat{\alpha}_0^{[l-1]}$, $\boldsymbol{\alpha}^{(0)} = \hat{\boldsymbol{\alpha}}^{[l-1]}$, $\mathbf{B}_1^{(0)} = \hat{\mathbf{B}}_1^{[l-1]}$, $\mathbf{B}_2^{(0)} = \hat{\mathbf{B}}_2^{[l-1]}, \ldots, \mathbf{B}_M^{(0)} = \hat{\mathbf{B}}_M^{[l-1]}$.
 5:   else
 6:     Initialize $\alpha_0^{(0)}$, $\boldsymbol{\alpha}^{(0)}$, $\mathbf{B}_1^{(0)}, \mathbf{B}_2^{(0)}, \ldots, \mathbf{B}_M^{(0)}$ from N(0,1) and set the first row of each $\mathbf{B}_m^{(0)}$, $m = 1, 2, \ldots, M-1$, to be one. Swap the columns of $\mathbf{B}_M^{(0)}$ to satisfy the inequality constraint in $S_{\mathcal{B}}$ and adjust $\mathbf{B}_1^{(0)}, \mathbf{B}_2^{(0)}, \ldots, \mathbf{B}_{M-1}^{(0)}$ in the same way with $\mathbf{B}_M^{(0)}$.
 7:   end if
 8:   Assemble $\boldsymbol{\gamma}^{(0)}$ according to (4.1).
 9:   Set $\tilde{\boldsymbol{\theta}}^{(0)} = \boldsymbol{\gamma}^{(1)} = \boldsymbol{\gamma}^{(0)}$. Let $\varrho_0 = 0$ and $\varrho_1 = 1$.
10:   for $t = 1, 2, \ldots$ do
11:     Let $\varrho_{t+1} = (1 + \sqrt{1 + 4\varrho_t^2})/2$.
12:     Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\gamma}^{(t)} + \frac{\varrho_{t-1}}{\varrho_t}(\tilde{\boldsymbol{\theta}}^{(t)} - \boldsymbol{\gamma}^{(t)}) + \frac{\varrho_{t-1}-1}{\varrho_t}(\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^{(t-1)})$.
13:     Compute $\tilde{d}_t = \|\tilde{\boldsymbol{\theta}}^{(t)} - \boldsymbol{\theta}^{(t-1)}\|_{\mathrm{F}}^2 / \langle \tilde{\boldsymbol{\theta}}^{(t)} - \boldsymbol{\theta}^{(t-1)}, \nabla C(\tilde{\boldsymbol{\theta}}^{(t)}) - \nabla C(\boldsymbol{\theta}^{(t-1)}) \rangle$.
14:     repeat
15:       $\tilde{d}_t \leftarrow \tilde{d}_t/2$.
16:     until $C(\tilde{\boldsymbol{\theta}}^{(t+1)}) \le C(\boldsymbol{\theta}^{(t)}) + \langle \nabla C(\boldsymbol{\theta}^{(t)}), \tilde{\boldsymbol{\theta}}^{(t+1)} - \boldsymbol{\theta}^{(t)} \rangle + \frac{1}{2\tilde{d}_t}\|\tilde{\boldsymbol{\theta}}^{(t+1)} - \boldsymbol{\theta}^{(t)}\|_{\mathrm{F}}^2$.
17:     Let $\tilde{\boldsymbol{\theta}}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \tilde{d}_t \nabla C(\boldsymbol{\theta}^{(t)})$.
18:     Compute $\bar{d}_t = \|\tilde{\boldsymbol{\gamma}}^{(t)} - \boldsymbol{\gamma}^{(t-1)}\|_{\mathrm{F}}^2 / \langle \tilde{\boldsymbol{\gamma}}^{(t)} - \boldsymbol{\gamma}^{(t-1)}, \nabla C(\tilde{\boldsymbol{\gamma}}^{(t)}) - \nabla C(\boldsymbol{\gamma}^{(t-1)}) \rangle$.
19:     repeat
20:       $\bar{d}_t \leftarrow \bar{d}_t/2$.
21:     until $C(\tilde{\boldsymbol{\gamma}}^{(t+1)}) \le C(\boldsymbol{\gamma}^{(t)}) + \langle \nabla C(\boldsymbol{\gamma}^{(t)}), \tilde{\boldsymbol{\gamma}}^{(t+1)} - \boldsymbol{\gamma}^{(t)} \rangle + \frac{1}{2\bar{d}_t}\|\tilde{\boldsymbol{\gamma}}^{(t+1)} - \boldsymbol{\gamma}^{(t)}\|_{\mathrm{F}}^2$.
22:     Let $\tilde{\boldsymbol{\gamma}}^{(t+1)} = \boldsymbol{\gamma}^{(t)} - \bar{d}_t \nabla C(\boldsymbol{\gamma}^{(t)})$.
23:     if $C(\tilde{\boldsymbol{\gamma}}^{(t+1)}) \le C(\tilde{\boldsymbol{\theta}}^{(t+1)})$ then
24:       $\boldsymbol{\gamma}^{(t+1)} = \tilde{\boldsymbol{\gamma}}^{(t+1)}$.
25:     else
26:       $\boldsymbol{\gamma}^{(t+1)} = \tilde{\boldsymbol{\theta}}^{(t+1)}$.
27:     end if
28:     Retrieve $\alpha_0^{(t+1)}$, $\boldsymbol{\alpha}^{(t+1)}$, $\mathbf{B}_1^{(t+1)}, \mathbf{B}_2^{(t+1)}, \ldots, \mathbf{B}_M^{(t+1)}$ from $\boldsymbol{\gamma}^{(t+1)}$ according to (4.1).
29:     if The KKT condition of problem (2.5) is satisfied then
30:       Swap the columns of $\mathbf{B}_M^{(t+1)}$ to satisfy the inequality constraint in $S_{\mathcal{B}}$ and adjust $\mathbf{B}_1^{(t+1)}, \mathbf{B}_2^{(t+1)}, \ldots, \mathbf{B}_{M-1}^{(t+1)}$ in the same way with $\mathbf{B}_M^{(t+1)}$.
31:       Let $\hat{\alpha}_0^{[l]} = \alpha_0^{(t+1)}$, $\hat{\boldsymbol{\alpha}}^{[l]} = \boldsymbol{\alpha}^{(t+1)}$, $\hat{\mathbf{B}}_1^{[l]} = \mathbf{B}_1^{(t+1)}$, $\hat{\mathbf{B}}_2^{[l]} = \mathbf{B}_2^{(t+1)}, \ldots, \hat{\mathbf{B}}_M^{[l]} = \mathbf{B}_M^{(t+1)}$.
32:       break
33:     end if
34:     if $l < 2$ then
35:       for k = 1, 2, 3, 4 do
36:         Repeat lines 6 - 33. Update the solution $\hat{\alpha}_0^{[1]}$, $\hat{\boldsymbol{\alpha}}^{[1]}$, $\hat{\mathbf{B}}_1^{[1]}, \hat{\mathbf{B}}_2^{[1]}, \ldots, \hat{\mathbf{B}}_M^{[1]}$ if the objective value is smaller.
37:       end for
38:     end if
39:   end for
40: end for
```

Lemma 1. *Let $\boldsymbol{\gamma}^\star$ be any accumulation point of the sequence $\{\boldsymbol{\gamma}^{(t)}\}$ obtained from Algorithm 1, then $\mathbf{0} \in \nabla F(\boldsymbol{\gamma}^\star)$.*

5. NUMERICAL STUDIES

5.1 Simulations

In this section, we examine the performance of DCT-SVM. We use extensive numerical experiments to perform the following tasks: (1) compare the performance of DCT-SVM with different choices of δ and as well as a data-driven δ from cross-validation; (2) study the performance of DCT-SVM using Gaussian and Epanechnikov kernels; (3) investigate the united framework of tensor large-margin classifiers with examples of the SVM, logistic regression, and Huberized SVM; (4) demonstrate the superior performance of DCT-SVM over popular state-of-the-art classifica-

Table 1. Example 1: the logit model. Classification error (in percentage) of large-margin tensor classifiers including DCT-SVM with $\delta = 1, 0.1, 0.01$ using Gaussian kernels (denoted by SVM$^{\mathrm{G}}_{\delta=1}$, SVM$^{\mathrm{G}}_{\delta=0.1}$, SVM$^{\mathrm{G}}_{\delta=0.01}$, respectively), tensor Huberized SVM (HSVM), tensor logistic regression (logit), DCT-SVM using a CV-tuned data-driven δ with Gaussian kernels (SVM$^{\mathrm{G}}_{\delta\text{-CV}}$), and data-driven DCT-SVM with Epanechnikov kernels (SVM$^{\mathrm{E}}_{\delta\text{-CV}}$). In each situation, the best method is marked by a black box. All the quantities are averaged over 50 runs and the standard errors are given in parentheses

n	p	SVM$^{\mathrm{G}}_{\delta=1}$	SVM$^{\mathrm{G}}_{\delta=0.1}$	SVM$^{\mathrm{G}}_{\delta=0.01}$	HSVM	logit	SVM$^{\mathrm{G}}_{\delta\text{-CV}}$	SVM$^{\mathrm{E}}_{\delta\text{-CV}}$
200	20	$\boxed{\textbf{14.31}}$	23.03	27.48	29.05	17.60	14.78	15.78
		(0.83)	(1.60)	(1.37)	(1.43)	(1.28)	(0.96)	(0.34)
	50	26.00	29.72	29.39	34.13	$\boxed{\textbf{25.54}}$	26.41	26.82
		(0.61)	(0.86)	(0.91)	(0.86)	(0.74)	(0.71)	(0.69)
	100	30.78	34.55	34.49	34.15	$\boxed{\textbf{29.21}}$	30.58	33.49
		(0.53)	(0.80)	(0.92)	(0.73)	(0.50)	(0.51)	(0.62)
400	20	$\boxed{\textbf{9.41}}$	13.42	19.36	19.99	10.97	$\boxed{\textbf{9.41}}$	10.26
		(0.60)	(1.29)	(1.64)	(1.75)	(1.14)	(0.60)	(0.21)
	50	$\boxed{\textbf{15.55}}$	15.97	16.40	19.31	15.77	$\boxed{\textbf{15.55}}$	15.78
		(0.28)	(0.49)	(0.58)	(0.65)	(0.30)	(0.28)	(0.34)
	100	20.68	22.05	21.57	22.11	$\boxed{\textbf{20.17}}$	20.68	21.70
		(0.31)	(0.44)	(0.45)	(0.43)	(0.30)	(0.31)	(0.46)
600	20	7.22	8.55	12.19	12.23	$\boxed{\textbf{6.55}}$	7.26	8.12
		(0.53)	(0.78)	(1.24)	(1.34)	(0.17)	(0.53)	(0.20)
	50	12.13	12.10	11.98	13.31	12.34	12.13	$\boxed{\textbf{11.87}}$
		(0.20)	(0.22)	(0.20)	(0.25)	(0.19)	(0.20)	(0.22)
	100	15.72	15.71	$\boxed{\textbf{15.59}}$	16.49	15.92	15.72	15.83
		(0.28)	(0.34)	(0.31)	(0.38)	(0.29)	(0.28)	(0.30)
800	20	$\boxed{\textbf{5.90}}$	7.24	10.31	9.60	5.97	5.91	6.86
		(0.44)	(0.61)	(1.20)	(1.07)	(0.47)	(0.45)	(0.14)
	50	9.57	9.57	9.75	9.93	$\boxed{\textbf{9.56}}$	9.57	9.85
		(0.15)	(0.14)	(0.15)	(0.14)	(0.16)	(0.15)	(0.14)
	100	$\boxed{\textbf{12.66}}$	12.93	12.82	13.01	13.18	$\boxed{\textbf{12.66}}$	12.85
		(0.23)	(0.36)	(0.35)	(0.29)	(0.24)	(0.23)	(0.31)
1000	20	$\boxed{\textbf{4.78}}$	5.82	9.75	6.64	4.86	$\boxed{\textbf{4.78}}$	6.70
		(0.12)	(0.18)	(1.10)	(0.13)	(0.15)	(0.12)	(0.17)
	50	8.34	9.03	8.94	9.42	8.27	8.34	$\boxed{\textbf{8.13}}$
		(0.13)	(0.54)	(0.53)	(0.53)	(0.14)	(0.13)	(0.17)
	100	10.68	$\boxed{\textbf{10.42}}$	10.72	10.62	11.43	10.68	11.09
		(0.20)	(0.25)	(0.24)	(0.27)	(0.17)	(0.20)	(0.27)

tion methods.

Example 1 In this example, simulated data are generated from the logistic regression model. In particular, for each $i = 1, 2, \ldots, n$, y_i is drawn from the Bernoulli distribution with the probability $P(y_i = 1) = 1/(1 + \exp(-\pi_i))$, where $\pi_i = \langle \mathcal{B}, \mathcal{X}_i \rangle + \mathbf{z}_i^\top \boldsymbol{\alpha} + 1$. Each element in $\mathcal{X}_i \in \mathbb{R}^{p\times 5\times 2}$ and $\mathbf{z}_i \in \mathbb{R}^p$ follows the standard Gaussian distribution. For the true parameters, the tensor coefficient $\mathcal{B} \in \mathbb{R}^{p\times 5\times 2}$ is given by

$$\mathcal{B} = \sum_{r=1}^{2} \boldsymbol{\beta}^{(1r)} \circ \boldsymbol{\beta}^{(2r)} \circ \boldsymbol{\beta}^{(3r)}, \tag{5.1}$$

where $\boldsymbol{\beta}^{(11)} \in \mathbb{R}^p$ has elements independently drawn from $\mathrm{N}(1, 0.05)$, $\boldsymbol{\beta}^{(12)} \in \mathbb{R}^p$ has elements generated from $\mathrm{N}(-2, 0.05)$, both $\boldsymbol{\beta}^{(21)} \in \mathbb{R}^5$ and $\boldsymbol{\beta}^{(22)} \in \mathbb{R}^5$ have elements from $\mathrm{N}(0, 1)$, $\boldsymbol{\beta}^{(31)} = (-1, 1)^\top$, and $\boldsymbol{\beta}^{(32)} = (-1, 0.5)^\top$; the vector-valued coefficient $\boldsymbol{\alpha} \in \mathbb{R}^p$ is given by independently generating each element from $\mathrm{N}(3, 1)$. In this example, we vary the sample size n from $\{200, 400, 600, 800, 1000\}$ and the dimension of the first tensor mode p from $\{20, 50, 100\}$. The Bayes error is around 2%.

We first fit our DCT-SVM using the Gaussian kernel with the bandwidth $\delta = 1, 0.1$, and 0.01. For each classifier, using five-fold cross-validation, we select the tensor rank R from $\{2, 3, 4\}$ and the shrinkage parameter λ from 10 candidate values that are uniformly distributed on the logarithm scale between 1 and 10^{-6}. The classification errors are summarized in Table 1. We see DCT-SVM with $\delta = 1$ is slightly better than the other two with marginal difference. This observation aligns with our theory that $\delta = O(1)$ suffices to allow the method to work. Yet despite the insensitive performance of DCT-SVM to δ, we propose a data-driven

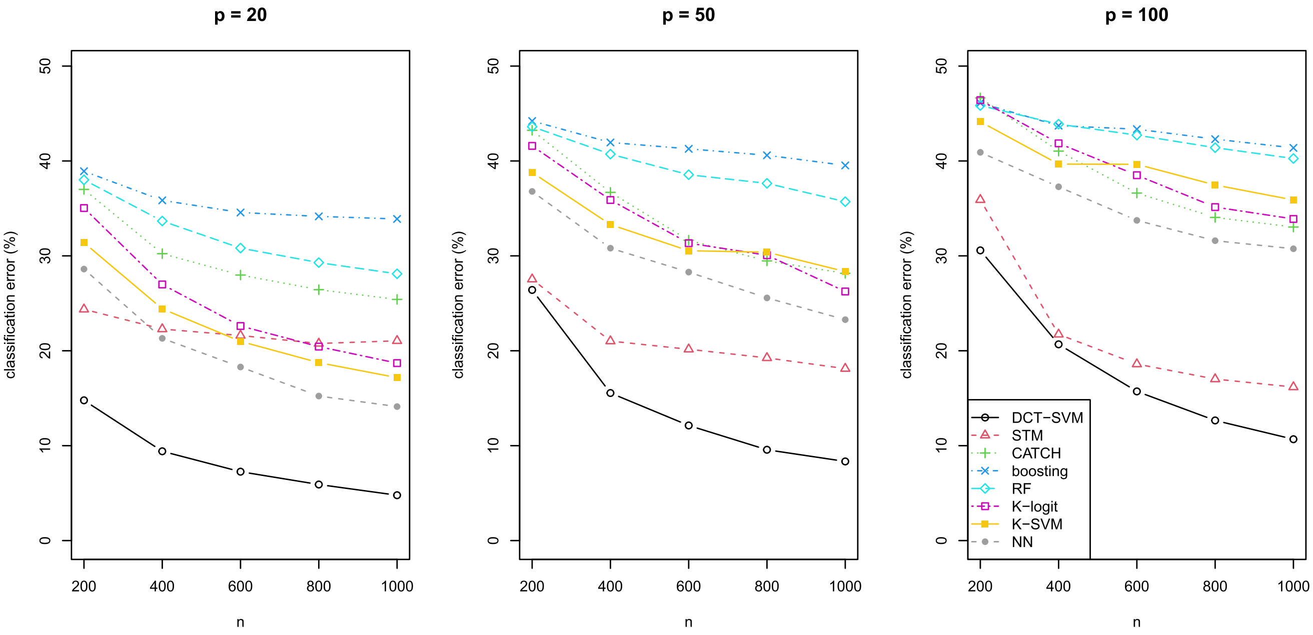

Figure 2. Example 1: the logit model. Classification error (in percentage) of DCT-SVM using the Gaussian kernel with a data-driven δ (denoted by DCT-SVM, which is the same method as $\mathrm{SVM}^{\mathrm{G}}_{\delta\text{-CV}}$ in Table 1), as compared with support tensor machines (STM), CATCH, boosting, random forest (denoted by RF), kernel logistic regression (denoted by K-logit), kernel SVM (denoted by K-SVM), and neural networks (denoted by NN).

strategy by treating δ as a hyperparameter to be selected by CV, and we see its performance is in par with the narrow winner, DCT-SVM with $\delta = 1$. Hence the data-driven strategy provides practitioners a relatively safe choice of δ. We further refit DCT-SVM with Epanechnikov kernels, whose error are higher than Gaussian kernels.

We then use DCT-SVM with the data-driven δ as an example and compare it with other baseline methods. We consider two tensor classifiers, the STM, using the Python library HOTTBOX [18], and CATCH, which extends linear discriminant analysis to tensor classification and is implemented in the R package `catch` [29]. These two methods directly classify the tensor data without vectorization, while STM cannot fit the vector-valued predictors, so we set its estimate of $\boldsymbol{\alpha}$ to be zero. We further unfold the tensor data into vectors and apply several popular vector-valued classifiers. We fit boosting with the exponential loss, i.e., the AdaBoost, using the R package `gbm` [12] with 500 decision trees and set the shrinkage parameter to be 0.01. We fit random forest using the R package `randomForest` [25] with 500 decision trees. We use the R package `magicsvm` [43] to fit both kernel logistic regression and kernel SVM, employing Gaussian kernels and using five-fold cross-validation to select the tuning parameter λ from 100 candidates ranging from 10^3 to 10^{-3}. We fit neural networks using the R package `nnet` [39] with two layers and ten units in the hidden layer. As shown in Figure 2, our DCT-SVM clearly outperforms all the baseline classifiers in this example.

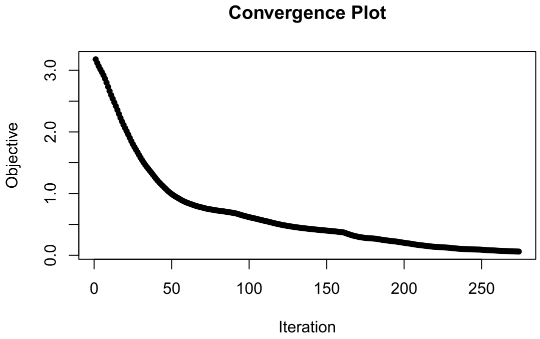

Figure 3. Convergence plot for solving a DCT-SVM model using Algorithm 1.

We further present a convergence plot to visualize the algorithmic convergence. We use Algorithm 1 to compute a DCT-SVM model for simulated data in Example 1 with $n = 600$ and $p = 50$. As shown in Figure 3, the objective value monotonically decreases.

Example 2 In this example, the tensor predictors from each class are generated from the tensor normal distribution:

$$\mathcal{X}|(y = k) \sim \mathrm{TN}(\mathcal{U}_k, \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ldots, \boldsymbol{\Sigma}_M), \ \ k = 1, 2,$$

that is, each $\mathcal{X}_i$ in the class k is independently generated from $\mathcal{U}_k + \mathcal{C} \times_1 \boldsymbol{\Sigma}_1^{1/2} \times_2 \boldsymbol{\Sigma}_2^{1/2} \ldots \times_M \boldsymbol{\Sigma}_M^{1/2}$, where $\mathcal{U}_k \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_M}$ is the mean tensor and $\mathcal{C} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_M}$

Table 2. Example 2: tensor normal distributions. Classification error (in percentage) of large-margin tensor classifiers including DCT-SVM with $\delta = 1, 0.1, 0.01$ using Gaussian kernels (denoted by SVM$^{\mathrm{G}}_{\delta=1}$, SVM$^{\mathrm{G}}_{\delta=0.1}$, SVM$^{\mathrm{G}}_{\delta=0.01}$, respectively), tensor Huberized SVM (HSVM), tensor logistic regression (logit), DCT-SVM using a CV-tuned data-driven δ with Gaussian kernels (SVM$^{\mathrm{G}}_{\delta\text{-CV}}$), and data-driven DCT-SVM with Epanechnikov kernels (SVM$^{\mathrm{E}}_{\delta\text{-CV}}$). In each situation, the best method is marked by a black box. All the quantities are averaged over 50 runs and the standard errors are given in parentheses

n	ρ	$\mathrm{SVM}^{\mathrm{G}}_{\delta=1}$	$\mathrm{SVM}^{\mathrm{G}}_{\delta=0.1}$	$\mathrm{SVM}^{\mathrm{G}}_{\delta=0.01}$	HSVM	logit	$\mathrm{SVM}^{\mathrm{G}}_{\delta\text{-CV}}$	$\mathrm{SVM}^{\mathrm{E}}_{\delta\text{-CV}}$
200	0.1	24.23	25.60	26.96	27.46	$\boxed{\mathbf{23.47}}$	24.19	27.83
		(0.66)	(0.63)	(0.47)	(0.39)	(0.69)	(0.66)	(0.69)
	0.5	27.69	28.53	28.66	28.03	$\boxed{\mathbf{27.47}}$	28.43	39.15
		(0.36)	(0.45)	(0.41)	(0.26)	(0.40)	(0.45)	(0.67)
	0.7	29.24	29.22	28.58	$\boxed{\mathbf{28.05}}$	28.18	29.37	42.83
		(0.54)	(0.46)	(0.35)	(0.27)	(0.40)	(0.52)	(0.46)
400	0.1	$\boxed{\mathbf{18.60}}$	22.19	24.16	23.97	20.04	18.83	20.47
		(0.76)	(0.80)	(0.70)	(0.73)	(0.85)	(0.78)	(0.28)
	0.5	$\boxed{\mathbf{25.19}}$	25.35	25.59	26.55	25.43	25.22	31.94
		(0.41)	(0.38)	(0.34)	(0.25)	(0.42)	(0.42)	(0.65)
	0.7	26.44	26.30	26.59	26.68	$\boxed{\mathbf{26.23}}$	26.33	38.96
		(0.26)	(0.26)	(0.27)	(0.24)	(0.32)	(0.25)	(0.56)
600	0.1	16.22	20.12	21.70	23.25	17.47	$\boxed{\mathbf{16.21}}$	18.26
		(0.69)	(0.82)	(0.80)	(0.72)	(0.87)	(0.69)	(0.22)
	0.5	23.92	24.77	24.77	25.77	$\boxed{\mathbf{23.64}}$	24.12	28.50
		(0.45)	(0.39)	(0.37)	(0.26)	(0.48)	(0.44)	(0.38)
	0.7	$\boxed{\mathbf{25.15}}$	25.70	25.84	26.20	25.63	25.33	36.95
		(0.24)	(0.24)	(0.23)	(0.21)	(0.23)	(0.23)	(0.65)
800	0.1	$\boxed{\mathbf{13.83}}$	17.44	19.34	18.70	16.50	14.13	17.39
		(0.46)	(0.88)	(0.92)	(0.88)	(0.82)	(0.52)	(0.15)
	0.5	$\boxed{\mathbf{22.63}}$	23.66	24.22	24.89	22.84	22.97	27.84
		(0.50)	(0.46)	(0.47)	(0.36)	(0.51)	(0.49)	(0.44)
	0.7	24.93	25.52	25.61	26.00	$\boxed{\mathbf{24.93}}$	25.05	34.99
		(0.30)	(0.27)	(0.25)	(0.22)	(0.31)	(0.30)	(0.66)
1000	0.1	$\boxed{\mathbf{14.78}}$	18.07	20.51	20.94	16.36	$\boxed{\mathbf{14.78}}$	17.11
		(0.65)	(0.90)	(0.93)	(0.91)	(0.82)	(0.65)	(0.17)
	0.5	$\boxed{\mathbf{23.52}}$	24.96	25.33	25.57	24.01	24.06	27.67
		(0.45)	(0.32)	(0.30)	(0.31)	(0.47)	(0.41)	(0.52)
	0.7	25.17	25.52	25.58	25.65	$\boxed{\mathbf{25.11}}$	25.37	33.92
		(0.27)	(0.23)	(0.25)	(0.23)	(0.30)	(0.26)	(0.56)

with each element drawn from the standard Gaussian distribution. The vector-valued predictor is given by

$$\mathbf{u}|(y = k) \sim \mathrm{N}_p(\boldsymbol{\mu}_k, \mathbf{I}_p), \ k = 1, 2.$$

We set $M = 3$ and $\mathcal{X} \in \mathbb{R}^{p\times 5\times 2}$, and define $\mathcal{B}$ in the same way with (5.1) and scale it to have the unit Forbenius norm. We let $\mathcal{U}_1 = \mathcal{B} \times_1 \boldsymbol{\Sigma}_1 \times_2 \boldsymbol{\Sigma}_2 \times_3 \boldsymbol{\Sigma}_3$ and set all elements in $\mathcal{U}_2$ to be zero. We let $\boldsymbol{\Sigma}_1$ have an autoregressive structure such that $(\boldsymbol{\Sigma}_1)_{i,j} = (-\rho)^{|i-j|}$, and let $\boldsymbol{\Sigma}_2$ and $\boldsymbol{\Sigma}_3$ have a composite symmetric structure with all diagonals to be 1 and off-diagonals 0.3. In this example, we fix $p = 20$ and vary ρ from $\{0.1, 0.5, 0.7\}$, giving the Bayes error as 14.0%, 23.4%, and 28.9%, respectively. We vary n from $\{200, 400, 600, 800, 1000\}$ and consider balanced classifications such that $\sum_{i=1}^{n} \mathbb{I}_{y_i=1} = n/2$.

We first fit tensor large-margin classifiers including DCT-SVM with $\delta = 1, 0.1, 0.01$ using Gaussian kernels, DCT-SVM with the data-driven δ using Gaussian and Epanechnikov kernels, tensor logistic regression, and tensor Huberized regression. From the classification errors exhibited in Table 2, we observe that the choice of δ in DCT-SVM does not contribute to significantly different classification accuracy, and the DCT-SVM with the data-driven δ performs similarly with $\delta = 1$. Gaussian kernels works better than Epanechnikov kernels for DCT-SVM. In addition, tensor logistic regression has the best performance in several cases; tensor Huberized SVM delivers the best classification when $\rho = 0.7$ and $n = 200$.

We further compare DCT-SVM with the data-driven δ with other state-of-the-art classifiers. From Figure 4, we see our proposed method yields the lowest error in all the situations except for $\rho = 0.1$ and $n = 200, 400$ where STM performs the best; overall the three tensor-based classifiers,

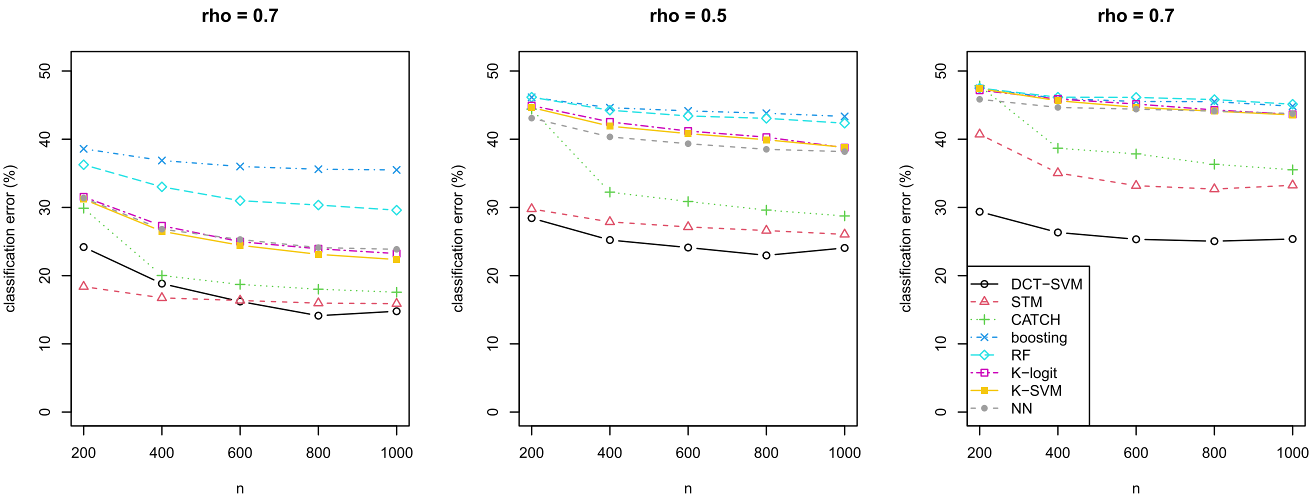

Figure 4. Example 2: tensor normal distributions. Classification error (in percentage) of DCT-SVM using the Gaussian kernel with a data-driven δ (denoted by DCT-SVM, which is the same method as SVM$^{\mathrm{G}}_{\delta\text{-}CV}$ in Table 2), as compared with support tensor machines (STM), CATCH, boosting, random forest (denoted by RF), kernel logistic regression (denoted by K-logit), kernel SVM (denoted by K-SVM), and neural networks (denoted by NN).

DCT-SVM, STM, and CATCH performs much better than the others. The simulation results reveal the great advantages of the tensor classification methods that exploit special tensor structures rather than resort to the vectorization.

5.2 A real application to online advertising

In this section, we study a real-world online advertising application. Online advertising is playing an essential role in attracting customers and securing ad revenues using the internet as a medium. As reported in [9], over 521 billion dollars were spent on online ads in 2021 worldwide; in 2026, the number is projected to be 876 billion dollars and the online ad spending will take about 75% of total media ad spending. For online advertising, typically, a successful ad conversion goes through the process of an impression, when the ad is displayed, a click, when the ad gets clicked on, and a conversion, when the ad eventually leads to a specified action such as a transaction or an email sign-up. To gauge the success of online ads, one of the most important metrics is the click-through rate (CTR), which is the ratio of the number of clicks to the number of impressions. CTR largely affects the revenue, as the revenue per one thousand impressions (RPM) is proportional to CTR multiplied by the cost per click. Also, it is our common belief that not all ads are effectively impacting the CTR [6, 47]. Hence it is crucial to building a model to study how the impressions on different ad campaigns affect the CTR, and it is profitable for online advertising companies to efficiently deliver ads to intended customers and maintain a high average CTR.

To enhance the performance of online advertising, we collected the information of 136 ad campaigns from a premium online advertising company. To protect private and sensitive information, all the reported data and results in this work are deliberately incomplete, anonymized, and are not related to the real portfolio of the company at any particular time. The selected 136 ad campaigns were randomly numbered with ID $1, 2, \ldots, 136$ without revealing any identifying information. We recorded the impression of the ad campaigns from 672 hours, and the users in each hour form a user group. We summarized the number of impressions and clicks of the campaigns delivered to each of the three devices, namely phones, tablets, and personal computers, and older and younger users that are partitioned by the median age of all the users. As a consequence, each user group has a tensor-valued impression $X_i \in \mathbb{R}^{136\times3\times2}$, $i = 1, 2, \ldots, 672$. For each user group, we calculated the overall CTR, the ratio of the total clicks to the total impressions, and our goal is to classify if the overall CTR of a user group is above the company-wise average (coded the response label as $+1$) or below that (coded as -1). By employing powerful classification algorithms to solve this classification problem, online advertising companies can optimize the ads delivery and thus make profits.

Among 672 user groups, we randomly selected 80% of them as the training data to train and tune all the classification methods, and reserved the remaining 20% of the user groups as the test data. We used the proposed DCT-SVM method with a Gaussian kernel and used five-fold cross-validation to select the tensor rank R from $\{2, 3, 4\}$, the regularization parameter λ from ten numbers uniformly distributed between 10^{-6} and 1 on the logarithm scale, and the data-driven δ from $\{1, 0.1, 0.01\}$. The performance of DCT-SVM is compared against all the baseline methods used in

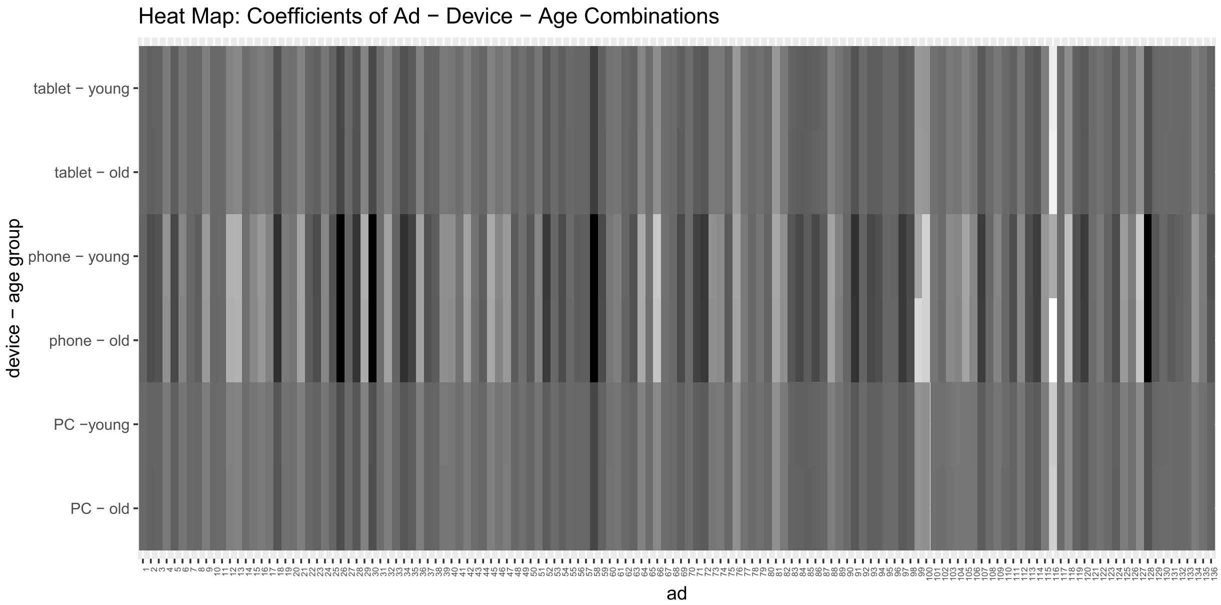

Figure 5. A heatmap of the coefficients of density-convoluted tensor SVM for classifying the overall CTR of an online advertising application. Each tile represents a tensor coefficient associated with an ad, device, and age group combination. The effect of each combination on the overall CTR is reflected by the darkness of the tiles. The IDs of all the ad campaigns have been intentionally renumbered for concerns of confidentiality.

the simulation studies. DCT-SVM gives the lowest classification error on the test data, 8.89%, which follows random forest with the test error 13.33% and adaBoost with 15.56%. The test errors of neural nets, kernel logistic regression, CATCH, and kernel SVM, are 20.00%, 20.74%, 22.96%, and 23.71%, respectively. In this example, the classification accuracy of DCT-SVM outperforms the vector-based baseline methods, most of which produce nonlinear classifiers.

Figure 5 shows a heatmap of the tensor coefficients to visualize the effect of each ad, device, and age group combination on the overall CTR. With all the 136 ad campaigns aligned in the horizontal axis and the six combinations of devices and age groups in the vertical axis, the darkness of the tiles depicts the corresponding tensor coefficients of the DCT-SVM model. In particular, a darker tile implies a larger change of the discriminant function, which tends to associate with a positive label, i.e., an overall CTR above the average. From Figure 5, we clearly observe the low-rank tensor structure, as the device and age group combinations share the same pattern of the coefficients of all the ad campaigns. The overall CTR is more sensitive to the ads being placed on phones than tablets and personal computers, while the ads delivered to users in different age groups do not elicit much different effects on the overall CTR. According to the linear nature of our DCT-SVM model, the overall CTR can be improved in the largest extent by adding the impression of the ad campaign with ID 58 or declining the campaign with ID 116 when they are delivered on phones.

6. DISCUSSION

In this paper, we have proposed a new classifier DCT-SVM for tensor data classification. DCT-SVM is motivated by smoothing the SVM loss to reduce the computational burden. Treating DCT-SVM as a new classifier rather than a computational remedy of tensor SVM and rigorously proved the convergence rate of risk using empirical process theory. We have developed an efficient algorithm and demonstrated the superior performance of DCT-SVM over many popular classifiers using extensive numerical studies.

In this work, we have employed CP decomposition to the tensor coefficients of the classifiers. It is known that Tucker decomposition is a more flexible low-rank tensor decomposition. It is interesting to extend the proposed methods with Tucker decomposition. In addition, future study can work towards the high-dimensional analysis of DCT-SVM, by imposing sparse penalties to select important variables from the tensor data. Further, due to the non-convexity of the objective function, Algorithm 1 does not guarantee converging to the global minimizer, and the strategy of multiple random initial starts has been adopted to address this issue. It will be interesting to investigate the global optimality of DCT-SVM through the landscape analysis [e.g., 23, 26].

A. APPENDIX: TECHNICAL PROOFS

Two lemmas for empirical processes are first presented and they are useful for proving Theorem 1.

Lemma 2. *(The symmetrization inequality; see for instance, Lemma 2.3.1 in [36]) Let $X_1, X_2, \dots, X_n$ be iid random variables defined on a probability space and let $\epsilon_1, \epsilon_2, \dots, \epsilon_n$ be iid Rademacher random variables, i.e., $P(\epsilon_i = 1) = P(\epsilon_i = -1) = 1/2$. With a class of measurable functions $\mathcal{H}$, it holds that*

$$\begin{aligned}&\mathbb{E}\left\{\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}|h(X_i)-\mathbb{E}[h(X_i)]|\right\}\\ \le&2\mathbb{E}\left\{\sup_{h\in\mathcal{H}}\left|\frac{1}{n}\sum_{i=1}\epsilon_i h(X_i)\right|\right\}.\end{aligned}$$

Lemma 3. *(The contraction inequality; see for instance, Theorem 4.12 in [20]) Let ϕ be a contraction, that is, $|\phi(s_1) - \phi(s_2)| \le |s_1 - s_2|$ for all $s_1, s_2 \in \mathbb{R}$ and $\phi(0) = 0$. Then with iid Rademacher random variables $\epsilon_1, \epsilon_2, \dots, \epsilon_n$ and a sequence of constants $s_1, s_2, \dots, s_n$, it holds that*

$$\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\phi(s_i)\right| \le 2\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\epsilon_i s_i\right|.$$

Proof of Theorem 1. By definition, we have

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^{n}\phi\left\{y_i\left(\hat\alpha_0+\mathbf{z}_i^\top\hat{\boldsymbol\alpha}+\langle\hat{\mathcal{B}},\mathcal{X}_i\rangle\right)\right\}\\ &\quad+\lambda\|\hat{\boldsymbol\alpha}\|_2^2+\lambda\left(\sum_{m=1}^{M}\|\hat{\mathbf{B}}_m\|^2\right)\\ \le&\frac{1}{n}\sum_{i=1}^{n}\phi\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}\\ &\quad+\lambda\|\boldsymbol\alpha^*\|_2^2+\lambda\left(\sum_{m=1}^{M}\|\mathbf{B}_m^*\|^2\right).\end{aligned}$$

By convexity of $\phi(\cdot)$ and mean value theorem, this further implies

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^{n}c_i''\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]^2\\ &+\frac{1}{n}\sum_{i=1}^{n}c_i'y_i\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]\\ \le&\lambda(\|\boldsymbol\alpha^*\|_2^2-\|\hat{\boldsymbol\alpha}\|_2^2)+\lambda\left(\sum_{m=1}^{M}\|\mathbf{B}_m^*\|^2-\|\hat{\mathbf{B}}_m\|^2\right),\end{aligned}$$

where

$$\begin{aligned}c_i''&=\phi''\left\{y_i\left(\tilde\alpha_0+\mathbf{z}_i^\top\tilde{\boldsymbol\alpha}+\langle\tilde{\mathcal{B}},\mathcal{X}_i\rangle\right)\right\},\\ c_i'&=\phi'\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\},\end{aligned}$$

for $i = 1, 2, \dots, n$, and

$$(\tilde\alpha_0,\tilde{\boldsymbol\alpha},\tilde{\mathcal{B}})=c_0(\hat\alpha_0,\hat{\boldsymbol\alpha},\hat{\mathcal{B}})+(1-c_0)(\alpha_0^*,\boldsymbol\alpha^*,\mathcal{B}^*)$$

with some $c_0 \in [0, 1]$.

By Assumption 1 and our setting, there exists a constant $C_0 > 0$ such that $\min_{1\le i\le n} c_i'' > C_0$. Consequently, (A.1) implies

$$\begin{aligned}&\frac{C_0}{n}\sum_{i=1}^{n}\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]^2\\ &+\frac{1}{n}\sum_{i=1}^{n}c_i'y_i\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]\\ \le&\lambda(\|\boldsymbol\alpha^*\|_2^2-\|\hat{\boldsymbol\alpha}\|_2^2)+\lambda\left(\sum_{m=1}^{M}\|\mathbf{B}_m^*\|^2-\|\hat{\mathbf{B}}_m\|^2\right).\end{aligned}$$

Define events

$$\begin{aligned}\mathcal{E}_1&:=\left\{\left|\frac{1}{n}\sum_{i=1}^{n}\phi'\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}y_i\right|\le\frac{\lambda}{2}\right\},\\ \mathcal{E}_2&:=\left\{\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}y_i\mathbf{z}_i\right\|\le\frac{\lambda}{2}\right\},\\ \mathcal{E}_3&:=\left\{\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}y_i\mathcal{X}_i\right\|\le\frac{\lambda}{2}\right\},\end{aligned}$$

and let $\mathcal{E} := \mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3$. We derive upper bounds for $\mathrm{P}(\mathcal{E}_1^{\mathrm{c}})$, $\mathrm{P}(\mathcal{E}_2^{\mathrm{c}})$, $\mathrm{P}(\mathcal{E}_3^{\mathrm{c}})$. Note that by definition of $(\alpha_0^*, \boldsymbol\alpha^*, \mathcal{B}^*)$, we have

$$\begin{aligned}&\mathbb{E}\left[\phi'\left\{y\left(\alpha_0^*+\mathbf{z}^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}\rangle\right)\right\}y\right]=0,\\ &\mathbb{E}\left[\phi'\left\{y\left(\alpha_0^*+\mathbf{z}^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}\rangle\right)\right\}y\mathbf{z}\right]=0,\\ &\mathbb{E}\left[\phi'\left\{y\left(\alpha_0^*+\mathbf{z}^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}\rangle\right)\right\}y\mathcal{X}\right]=0.\end{aligned}$$

By Hoeffding's inequality, we first have

$$\text{(A.1)}\qquad \mathrm{P}(\mathcal{E}_1^{\mathrm{c}})\le 2\exp\left\{-\frac{n\lambda^2}{8}\right\}.$$

Next, we upper bound $\mathrm{P}(\mathcal{E}_2^{\mathrm{c}})$. Note that for any $\mathbf{v} \in \mathbb{R}^{p_0}$ such that $\|\mathbf{v}\| \le 1$; since $|\phi'(\cdot)| \le 1$,

$$\begin{aligned}&\mathrm{Var}\left(\phi'\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}y\mathbf{z}^\top\mathbf{v}\right)\\ \le&\mathbb{E}[(\mathbf{z}^\top\mathbf{v})^2]\le\Pi^2.\end{aligned}$$

Let $\sigma_1, \dots, \sigma_n$ be i.i.d. Rademacher random variables (i.e., $\mathrm{P}(\sigma_i = 1) = \mathrm{P}(\sigma_i = -1) = 1/2$), which are independent from all the other random elements. Now, by Lemma 2.3.7 in [36], bounded differences inequality (for instance see Corollary 2.21 in [40]), we have

$$\begin{aligned}\mathrm{P}(\mathcal{E}_2^{\mathrm{c}})&=\mathrm{P}\left(\sup_{\mathbf{v}\in\mathbb{R}^{p_0}:\|\mathbf{v}\|\le1}\left|\frac{1}{n}\sum_{i=1}^{n}c_i'y_i\mathbf{z}_i^\top\mathbf{v}\right|>\frac{\lambda}{2}\right)\\ &\le\frac{2}{1-\frac{16\Pi^2}{n\lambda^2}}\mathrm{P}\left(\left\|\frac{1}{n}\sum_{i=1}^{n}\sigma_i c_i'y_i\mathbf{z}_i\right\|>\frac{\lambda}{8}\right)\end{aligned}$$

$$\begin{aligned}&\le \frac{2}{1-\frac{16\Pi^2}{n\lambda^2}}\mathbb{E}\left[2\exp\left\{-\frac{n^2\lambda^2}{128\sum_{i=1}^n\|\mathbf{z}_i\|^2}\right\}\right]\\ &\le \frac{4}{1-\frac{16\Pi^2}{n\lambda^2}}\exp\left\{-\frac{n\lambda^2}{128\Pi^2}\right\}. \tag{A.2}\end{aligned}$$

The bound for $\mathrm{P}(\mathcal{E}_3^c)$ can be obtained with the same technique, i.e., we have

$$\mathrm{P}(\mathcal{E}_3^c) \le \frac{4}{1-\frac{16\Pi^2}{n\lambda^2}}\exp\left\{-\frac{n\lambda^2}{128\Pi^2}\right\}. \tag{A.3}$$

Combining (A.1), (A.2), and (A.3) and applying an union bound, we have

$$\mathrm{P}(\mathcal{E}^c) \le 2\exp\left\{-\frac{n\lambda^2}{8}\right\} + \frac{8}{1-\frac{16\Pi^2}{n\lambda^2}}\exp\left\{-\frac{n\lambda^2}{128\Pi^2}\right\}. \tag{A.4}$$

Under $\mathcal{E}$, (A.1) implies

$$\begin{aligned}&\frac{C_0}{n}\sum_{i=1}^n\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]^2\\ &-\frac{\lambda}{2}\left[|\hat\alpha_0-\alpha_0^*|+\|\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*\|+\|\hat{\mathcal{B}}-\mathcal{B}^*\|\right]\\ \le&\lambda(\|\boldsymbol\alpha^*\|_2^2-\|\hat{\boldsymbol\alpha}\|_2^2)+\lambda\left(\sum_{m=1}^M\|\mathbf{B}_m^*\|^2-\|\hat{\mathbf{B}}_m\|^2\right),\end{aligned}$$

and then the above inequality implies that there exists some constant $G_0>0$ such that under $\mathcal{E}$,

$$\frac{C_0}{n}\sum_{i=1}^n\left[(\hat\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\hat{\boldsymbol\alpha}-\boldsymbol\alpha^*)+\langle\hat{\mathcal{B}}-\mathcal{B}^*,\mathcal{X}_i\rangle\right]^2\le\lambda G_0. \tag{A.5}$$

Define $\ell(\alpha_0,\boldsymbol\alpha,\mathcal{B})=\frac{1}{n}\sum_{i=1}^n\phi\left\{y_i\left(\alpha_0+\mathbf{z}_i^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}_i\rangle\right)\right\}$ for any $(\alpha_0,\boldsymbol\alpha,\mathcal{B})$. Also, define

$$\mathbb{D}=\left\{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\Theta:\frac{C_0}{n}\sum_{i=1}^n c_{1i}^2\le\lambda G_0\right\},$$

where for $i=1,2,\dots,n$,

$$c_{i1}=(\alpha_0-\alpha_0^*)+\mathbf{z}_i^\top(\boldsymbol\alpha-\boldsymbol\alpha^*)+\langle\mathcal{B}-\mathcal{B}^*,\mathcal{X}_i\rangle.$$

Let $G(\alpha_0,\boldsymbol\alpha,\mathcal{B})=\ell(\alpha_0,\boldsymbol\alpha,\mathcal{B})-\ell(\alpha_0^*,\boldsymbol\alpha^*,\mathcal{B}^*)$. Let $\mathcal{A}$ be the sub-σ field generated by $\{\mathbf{z}_i,\mathcal{X}_i\}_{i=1}^n$.

Let

$$X_1=\sup_{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\mathbb{D}}\left|G(\alpha_0,\boldsymbol\alpha,\mathcal{B})-\mathbb{E}[G(\alpha_0,\boldsymbol\alpha,\mathcal{B})|\mathcal{A}]\right|.$$

We give an upper bound for $\mathbb{E}[X_1]$. Recall that $\sigma_1,\dots,\sigma_n$ are i.i.d. Rademacher random variables which are independent from all the other random elements. By the symmetrization inequality (Lemma 2) and the contraction inequality (Lemma 3), $|\phi'(\cdot)|\le 1$ and Cauchy-Schwarz inequality, we have

$$\begin{aligned}\mathbb{E}[X_1]=&\mathbb{E}[\mathbb{E}[X_1|\mathcal{A}]]\\ \le&2\mathbb{E}\left[\sup_{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\mathbb{D}}\left|\frac{1}{n}\sum_{i=1}^n\sigma_i\left[\phi\left\{y_i\left(\alpha_0+\mathbf{z}_i^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}_i\rangle\right)\right\}\right.\right.\right.\\ &\left.\left.\left.-\phi\left\{y_i\left(\alpha_0^*+\mathbf{z}_i^\top\boldsymbol\alpha^*+\langle\mathcal{B}^*,\mathcal{X}_i\rangle\right)\right\}\right]\right|\right]\\ \le&4\mathbb{E}\left[\sup_{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\mathbb{D}}\left|\frac{1}{n}\sum_{i=1}^n\sigma_ic_{i1}\right|\right]\\ \le&4\mathbb{E}\left[\sup_{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\mathbb{D}}\sqrt{\frac{1}{n}\sum_{i=1}^n c_{i1}^2}\right]\\ \le&4\sqrt{\lambda\frac{G_0}{C_0}},\end{aligned}$$

where the last step is by the definition of $\mathbb{D}$. This implies that $X_1=O_p(\sqrt{\lambda})$.

Next, let

$$X_2=\sup_{(\alpha_0,\boldsymbol\alpha,\mathcal{B})\in\mathbb{D}}\Big|\mathbb{E}[G(\alpha_0,\boldsymbol\alpha,\mathcal{B})|\mathcal{A}]-\mathbb{E}[G(\alpha_0,\boldsymbol\alpha,\mathcal{B})]\Big|.$$

We give an upper bound for $\mathbb{E}[X_2]$. Let

$$\Phi(\mathbf{z},\mathcal{X},\alpha_0,\boldsymbol\alpha,\mathcal{B}):=\mathbb{E}\Big[\phi\{y(\alpha_0+\mathbf{z}^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}\rangle)\}\Big|\mathbf{z},\mathcal{X}\Big],$$

then

$$\begin{aligned}&\mathbb{E}[G(\alpha_0,\boldsymbol\alpha,\mathcal{B})|\mathcal{A}]\\ =&\frac{1}{n}\sum_{i=1}^n\left(\Phi(\mathbf{z}_i,\mathcal{X}_i,\alpha_0,\boldsymbol\alpha,\mathcal{B})-\Phi(\mathbf{z}_i,\mathcal{X}_i,\alpha_0^*,\boldsymbol\alpha^*,\mathcal{B}^*)\right)\end{aligned}$$

which again can be viewed as an empirical process. Notice that $\Phi(\cdot)$ depends on $(\alpha_0,\boldsymbol\alpha,\mathcal{B})$ only through the value of $\alpha_0+\mathbf{z}^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}\rangle$. Meanwhile, since $|\phi'(\cdot)|\le 1$, for any $(\alpha_0,\boldsymbol\alpha,\mathcal{B})$ and $(\alpha_0',\boldsymbol\alpha',\mathcal{B}')$, we have

$$\begin{aligned}&|\Phi(\mathbf{z},\mathcal{X},\alpha_0,\boldsymbol\alpha,\mathcal{B})-\Phi(\mathbf{z},\mathcal{X},\alpha_0',\boldsymbol\alpha',\mathcal{B}')|\\ =&\Big|\mathbb{E}\Big[\phi\{y(\alpha_0+\mathbf{z}^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}\rangle)\}\\ &-\phi\{y(\alpha_0'+\mathbf{z}^\top\boldsymbol\alpha'+\langle\mathcal{B}',\mathcal{X}\rangle)\}\Big|\mathbf{z},\mathcal{X}\Big]\Big|\\ \le&|(\alpha_0+\mathbf{z}^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}\rangle)-(\alpha_0'+\mathbf{z}^\top\boldsymbol\alpha'+\langle\mathcal{B}',\mathcal{X}\rangle)|,\end{aligned}$$

which means Φ is Lipschitz in the value of $\alpha_0+\mathbf{z}^\top\boldsymbol\alpha+\langle\mathcal{B},\mathcal{X}\rangle$. Again, by the symmetrization inequality, the contraction inequality, and Cauchy-Schwarz inequality, we have

$$\begin{aligned}\mathbb{E}[X_2] \le& 2\mathbb{E}\Bigg[\sup_{(\alpha_0,\boldsymbol{\alpha},\mathcal{B})\in\mathbb{D}}\Bigg|\frac{1}{n}\sum_{i=1}^{n}\sigma_i\Big[\Phi(\mathbf{z}_i,\mathcal{X}_i,\alpha_0,\boldsymbol{\alpha},\mathcal{B}) \\ &\quad - \Phi(\mathbf{z}_i,\mathcal{X}_i,\alpha_0^*,\boldsymbol{\alpha}^*,\mathcal{B}^*)\Big]\Bigg|\Bigg] \\ \le& 4\mathbb{E}\Bigg[\sup_{(\alpha_0,\boldsymbol{\alpha},\mathcal{B})\in\mathbb{D}}\Bigg|\frac{1}{n}\sum_{i=1}^{n}\sigma_i c_{i1}\Bigg|\Bigg] \\ \le& 4\mathbb{E}\Bigg[\sup_{(\alpha_0,\boldsymbol{\alpha},\mathcal{B})\in\mathbb{D}}\sqrt{\frac{1}{n}\sum_{i=1}^{n}c_{i1}^2}\Bigg] \\ \le& 4\sqrt{\lambda\frac{G_0}{C_0}}.\end{aligned}$$

This implies that $X_2 = O_p(\sqrt{\lambda})$.

Following (A.1), we know there exists some constant $C_1 > 0$ such that

$$G(\hat{\alpha}_0,\hat{\boldsymbol{\alpha}},\hat{\mathcal{B}}) \le C_1\lambda. \tag{A.6}$$

Since $\mathcal{R}(\alpha_0,\boldsymbol{\alpha},\mathcal{B}) = \mathbb{E}[G(\alpha_0,\boldsymbol{\alpha},\mathcal{B})]$, by triangle inequality, (A.5), and (A.6), we have under $\mathcal{E}$,

$$\mathcal{R}(\hat{\alpha}_0,\hat{\boldsymbol{\alpha}},\hat{\mathcal{B}}) \le X_1 + X_2 + C_1\lambda = O_p(\sqrt{\lambda}+\lambda).$$

Now, for any $\epsilon > 0$, the above implies that there exists a large enough constant $T_1 > 0$ such that

$$\limsup_{n\to\infty} \mathrm{P}\left\{\mathcal{R}(\hat{\alpha}_0,\hat{\boldsymbol{\alpha}},\hat{\mathcal{B}}) > T_1(\sqrt{\lambda}+\lambda)\right\} < \frac{\epsilon}{2} + \mathrm{P}(\mathcal{E}^c). \tag{A.7}$$

Choosing $\lambda = \sqrt{T_2/n}$ with large enough $T_2 > 0$ in (A.4), we have $\mathrm{P}(\mathcal{E}^c) < \epsilon/2$. Then (A.7) gives

$$\limsup_{n\to\infty} \mathrm{P}\left\{\mathcal{R}(\hat{\alpha}_0,\hat{\boldsymbol{\alpha}},\hat{\mathcal{B}}) > T_1\left(\left(\frac{T_2}{n}\right)^{\frac{1}{4}} + \sqrt{\frac{T_2}{n}}\right)\right\} < \epsilon. \tag{A.8}$$

By the arbitrariness of ϵ and the order of magnitude of n in the probability, (A.8) implies $\mathcal{R}(\hat{\alpha}_0,\hat{\boldsymbol{\alpha}},\hat{\mathcal{B}}) = O_p(n^{-\frac{1}{4}})$. The proof is finished. □

Proof of Theorem 2. We first give some general formula regarding the loss function $\phi = \phi_\delta$ and its derivatives. Direct calculation gives

$$\begin{aligned}\phi_\delta(t) &= \int_{-\infty}^{1}\frac{1-u}{\delta}K\left(\frac{t-u}{\delta}\right)\mathrm{d}u, \\ \phi'_\delta(t) &= -\int_{-\infty}^{\frac{1-t}{\delta}}K(u)\mathrm{d}u, \\ \phi''_\delta(t) &= \frac{1}{\delta}K\left(\frac{1-t}{\delta}\right), \ \forall t\in\mathbb{R}.\end{aligned}$$

It is straightforward to show that $\lim_{t\to\infty}\phi_\delta(t) = 0$ and $\lim_{t\to-\infty}\phi_\delta(t) = \infty$. We compute the $\psi(\cdot)$ function and the $H(\cdot)$ function defined in [1]. Since $\phi_\delta(\cdot)$ is convex and $\phi'_\delta(0) = -\int_{-\infty}^{\frac{1}{\delta}}K(u)\mathrm{d}u < 0$, by Theorem 2 of [1], $\psi(\theta) = \phi_\delta(0) - H(\frac{1+\theta}{2})$. For $H(\cdot)$, by definition in [1],

$$H(\eta) = \inf_{\alpha\in\mathbb{R}}\big(\eta\phi_\delta(\alpha) + (1-\eta)\phi_\delta(-\alpha)\big)$$

for $\eta\in[0,1]$. Let $\alpha_\eta = \operatorname{argmin}_{\alpha\in\mathbb{R}}\big(\eta\phi_\delta(\alpha)+(1-\eta)\phi_\delta(-\alpha)\big)$, whose existence is guaranteed by convexity of $\phi_\delta(\cdot)$ and the fact that $\phi_\delta(t)\to\infty$ as $t\to-\infty$. So

$$H(\eta) = \eta\phi_\delta(\alpha_\eta) + (1-\eta)\phi_\delta(-\alpha_\eta). \tag{A.9}$$

Meanwhile, by optimality condition,

$$\eta\phi'_\delta(\alpha_\eta) = (1-\eta)\phi'_\delta(-\alpha_\eta). \tag{A.10}$$

When $\eta = \frac{1}{2}$, the above equation is reduced to $\phi'_\delta(\alpha_{1/2}) = \phi'_\delta(-\alpha_{1/2})$ and we get $\alpha_{1/2} = 0$. We now compute $\psi(0)$, $\psi'(0)$ and $\psi''(0)$. By definition $\psi'(\theta) = -\frac{1}{2}H'(\frac{1+\theta}{2})$ and $\psi''(\theta) = -\frac{1}{4}H''(\frac{1+\theta}{2})$. Meanwhile, let α'_η be the derivative of α_η with respect to η, we have

$$\begin{aligned}&H'(\eta) \\ =&\phi_\delta(\alpha_\eta) + \eta\phi'_\delta(\alpha_\eta)\alpha'_\eta - \phi_\delta(-\alpha_\eta) - (1-\eta)\phi'_\delta(-\alpha_\eta)\alpha'_\eta \\ =&\phi_\delta(\alpha_\eta) - \phi_\delta(-\alpha_\eta),\end{aligned}$$

where the last step is by (A.10), and we have

$$H''(\eta) = \phi'_\delta(\alpha_\eta)\alpha'_\eta + \phi'_\delta(-\alpha_\eta)\alpha'_\eta. \tag{A.11}$$

Taking derivative with respect to η on both sides of (A.10), we have

$$\phi'_\delta(\alpha_\eta) + \eta\phi''_\delta(\alpha_\eta)\alpha'_\eta = -\phi'_\delta(-\alpha_\eta) - (1-\eta)\phi''_\delta(-\alpha_\eta)\alpha'_\eta,$$

which means

$$\alpha'_\eta = \frac{-\phi'_\delta(\alpha_\eta) - \phi'_\delta(-\alpha_\eta)}{\eta\phi''_\delta(\alpha_\eta) + (1-\eta)\phi''_\delta(-\alpha_\eta)}.$$

Plugging this into (A.11) we have

$$H''(\eta) = -\frac{\big(\phi'_\delta(\alpha_\eta) + \phi'_\delta(-\alpha_\eta)\big)^2}{\eta\phi''_\delta(\alpha_\eta) + (1-\eta)\phi''_\delta(-\alpha_\eta)}. \tag{A.12}$$

Hence combining (A.9), (A.11) and (A.12), we have

$$\begin{aligned}\psi(0) &= \phi_\delta(0) - H\left(\frac{1}{2}\right) \\ &= \phi_\delta(0) - \frac{1}{2}\phi_\delta(\alpha_{1/2}) - \frac{1}{2}\phi_\delta(\alpha_{1/2}) = 0, \\ \psi'(0) &= -\frac{1}{2}H'\left(\frac{1}{2}\right) = -\frac{1}{2}(\phi_\delta(0) - \phi_\delta(0)) = 0, \\ \psi''(0) &= -\frac{1}{4}H''\left(\frac{1}{2}\right) = \frac{\phi'_\delta(0)^2}{\phi''_\delta(0)} > 0.\end{aligned}$$

Therefore, we have

$$\psi(t) = \frac{\phi'_\delta(0)^2}{2\phi''_\delta(0)} t^2 + o(t^2) \tag{A.13}$$

where $o(t^2)$ is negligible compared with t^2 as $t \to 0$. Moreover, since $H''(\cdot) < 0$ by (A.12), $H'(\cdot)$ is a strictly decreasing function on $(0,1)$. Also, $H'(1/2) = 0$, so $H'(\cdot)$ is positive on $(0, 1/2)$ and negative on $(1/2, 1)$, which means $H(\cdot)$ is increasing on $(0, 1/2)$ and decreasing on $(1/2, 1)$. Thus, $\psi(\cdot)$ is decreasing on $(-1, 0)$ and increasing on $(0, 1)$. On the other hand, from (A.10), we know as $\eta \to 0$, $\alpha_\eta \to -\infty$, and as $\eta \to 1$, $\alpha_\eta \to \infty$. As a result, we have $\lim_{\theta \to -1} \psi(\theta) = \lim_{\theta \to 1} \psi(\theta) = \phi_\delta(0)$. Combining this and (A.13), it is straightforward to see that $\inf_{t \in (-1,1)} \psi(t)/t^2 > 0$. By Theorem 1 of [1] we have

$$\psi\big(\mathcal{R}_{0-1}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}})\big) \leq \mathcal{R}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}),$$

so combining this and previous result we have

$$\mathcal{R}^2_{0-1}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}) \lesssim \mathcal{R}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}).$$

Thus by our Theorem 1, we get $\mathcal{R}_{0-1}(\hat{\alpha}_0, \hat{\boldsymbol{\alpha}}, \hat{\mathcal{B}}) = O_p(n^{-\frac{1}{8}})$. □

ACKNOWLEDGEMENTS

The authors sincerely thank the editors and referees for their helpful comments and suggestions.

Received 1 October 2022

REFERENCES

[1] Bartlett, P.L., Jordan, M.I. and McAuliffe, J.D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, **101**(473), 138–156. MR2268032

[2] Barzilai, J. and Borwein, J.M. (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, **8**(1), 141–148. MR0967848

[3] Bertsekas, D. P. (1973). Stochastic optimization problems with nondifferentiable cost functionals. *Journal of Optimization Theory and Applications* **12**(2), 218–231. MR0329725

[4] Bi, X., Tang, X., Yuan, Y., Zhang, Y., and Qu, A. (2021). Tensors in statistics. *Annual Review of Statistics and Its Application*, **8**, 345–368. MR4243551

[5] Chi, E. C. and Kolda, T. G. (2012). On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, **33**, 1272–1299. MR3023474

[6] Choi, Y., Fontoura, M., Gabrilovich, E., Josifovski, V., Mediano, M., and Pang, B. (2010). Using landing pages for sponsored search ad selection. *Proceedings of the 19th International Conference on World Wide Web*, 251–260.

[7] Cook, R. D. and Li, B. and Chiaromonte, F. (2010). Envelope models for parsimonious and efficient multivariate linear regression. *Statistica Sinica* **20** 927–960. MR2729839

[8] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning* **20** 273–297.

[9] Cramer-Flood, E. (2022). Worldwide Ad Spending 2022. *Insider Intelligence.* https://www.insiderintelligence.com/content/worldwide-ad-spending-2022 (assessed on Sep 30, 2022).

[10] Fernandes, M., Guerre, E. and Horta, E. (2021). Smoothing quantile regressions. *Journal of Business and Economic Statistics* **39**(1), 338–357. MR4187194

[11] Fernández-Delgado, M., Cernadas, E., Barro, S. and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research* **15** 3133–3181. MR3277155

[12] Greenwell, B., Boehmke, B., Cunningham, J., and GBM Developers (2020). gbm: Generalized Boosted Regression Models. R package version 2.1.8.

[13] Hao, Z, He, L., Chen, B., and Yang, X. (2013). A linear support higher-order tensor machine for classification. *IEEE Transcations on Image Processing*, **22**(7), 2911–2920.

[14] Hao, B., Wang, B., Wang, P., Zhang, J., Yang, J., and Sun, W. (2021). Sparse tensor additive regression. *Journal of Machine Learning Research*, **22**(64), 1–43. MR4253757

[15] Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning: Prediction, Inference, and Data Mining*, 2nd edition, Springer, New York. MR2722294

[16] He, X., Pan, X., Tan, K. M. and Zhou, W.-X. (2021). Smoothed quantile regression with large-scale inference. *Journal of Econometrics*, **232**(2), 367–388. MR4539491

[17] Kiers, H.A.L. (2000). Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, **14**(3), 105–122.

[18] Kisil, I., Calvi, G.G., Dees, B.S., and Mandic, D.P. (2000). HOTTBOX: Higher Order Tensor ToolBOX. *arXiv preprint* arXiv:2111.15662.

[19] Kolda, T. and Bader, B. (2009). Tensor decompositions and applications. *SIAM Review*, **51**(3), 455–500. MR2535056

[20] Ledoux, M. and Talagrand, M. (2013). *Probability in Banach Spaces: Isoperimetry and Processes.* Springer Science & Business Media. MR2814399

[21] Li, H. and Lin, Z. (2015). Accelerated proximal gradient methods for nonconvex programming. *Advances in Neural Information Processing Systems*, 379–387.

[22] Li, Q. and Schonfeld, D. (2014). Multilinear discriminant analysis for higher-order tensor data classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36**(12), 2524–2537.

[23] Luo, Y., Li, X., and Zhang, A. (2022). Nonconvex factorization and manifold formulations are almost equivalent in low-rank matrix optimization. *arXiv preprint* arXiv:2108.01772.

[24] Li, X. Xu, D., Zhou, H., and Li, L. (2018). Tucker tensor regression and neuroimaging analysis. *Statistics in Biosciences*, **10**, 520–545.

[25] Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News* **2**, 18–22.

[26] Mei, S., Bai, Y., and Montanari, A. (2018). The landscape of empirical risk for nonconvex losses. *The Annals of Statistics*, **46**(6), 2747–2774. MR3851754

[27] Molstad, A. J. and Rothman, A. J. (2019). A penalized likelihood method for classification with matrix-valued predictors. *Journal of Computational and Graphical Statistics* **28**(1), 11–22. MR3939368

[28] Nesterov, Y. (2018), *Lectures on Convex Optimization*, Springer. MR3839649

[29] Pan, Y., Mai, Q., and Zhang, X. (2019). Covariate-adjusted tensor classification in high dimensions. *Journal of American Statistical Association*, **114**(527), 1305–1319. MR4011781

[30] Rubinstein, R. Y. (1983). Smoothed functionals in stochastic optimization. *Mathematics of Operations Research* **8**(1), 26–33. MR0703823

[31] Sun, W. W., Lu, J., Liu, H., and Cheng, G. (2017). Provable sparse tensor decomposition. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **79**(3), 899–916. MR3641413

[32] Sun, W.W., Hao, B., and Li, L. (2021). Tensors in modern statistical learning. *Wiley StatsRef: Statistics Reference Online*, 1–25.

[33] Tan, K.M., Wang, L., and Zhou, W.X. (2022). High-dimensional quantile regression: convolution smoothing and concave regularization. *Journal of the Royal Statistical Society, Series B*, **84**(1), 205–233. MR4400395

[34] Tao, D., Li, X., Hu, W., Maybank, S., and Wu, X. (2005). Support tensor learning. *Proceedings of the Fifth IEEE International Conference on Data Mining*, 1–8.

[35] Tucker, L.R. (1966). Some mathematical notes on three-mode factor analysis *Psychometrika*, **31**(3), 279–311. MR0205395

[36] Van Der Vaart, A. W. and Wellner, J. A. (1996). *Weak Convergence. In Weak Convergence and Empirical Processes.* Springer. MR1385671

[37] Vapnik, V. (1995). *The Nature of Statistical Learning Theory.* Springer Science & Business Media. MR1367965

[38] Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 988–999.

[39] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S.* New York: Springer, 4th ed. MR1337030

[40] Wainwright, M. J. (2019). *High-Dimensional Statistics: A Non-Asymptotic Viewpoint.* Cambridge University Press. MR3967104

[41] Wald, A. (1949). Note on the consistency of the maximum likelihood estimate. *The Annals of Mathematical Statistics*, **20**(4), 595–601 MR0032169

[42] Wang, B. Zhou, L, Gu, Y., and Zou, H. (2022). Density-convoluted support vector machines for high-dimensional classification. *IEEE Transactions on Information Theory*, **69**(4), 2523–2536. MR4570514

[43] Wang, B. and Zou, H. (2022). Fast and exact cross-validation theory for large-margin classification. *Technometrics*, **64**(3), 291–298. MR4457323

[44] Wang, N., Wang, W., and Zhang, X. (2022). Parsimonious tensor discriminant analysis. *Statistica Sinica*, in press.

[45] Wimalawarne, K., Tomioka, R. and Sugiyama, M. (2016). Theoretical and experimental analyses of tensor-based regression and classification. *Neural Computation*, **28**, 686–715. MR3867768

[46] Xia, D., Zhang, A., and Zhou, Y. (2022). Inference for low-rank tensors – no need to debias. *The Annals of Statistics*, **50**(2), 1220–1245. MR4404934

[47] Xu, J., Shao, X., Ma, J., Lee, K. C., Qi, H., and Lu, Q. (2016). Lift-based bidding in ad selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, **30**, 651–657.

[48] Zhang, A., Luo, Y., Raskutti, G., and Yuan, M. (2020) ISLET: Fast and optimal low-rank tensor regression via importance sketching. *SIAM Journal on Mathematics of Data Science*, **2**(2), 444–479. MR4106613

[49] Zhang, A. and Xia, D. (2018). Tensor SVD: Statistical and computational limits. *IEEE Transactions on Information Theory* **64**(11), 7311–7338. MR3876445

[50] Zhong, W. and Suslick, K. S. (2015). Matrix discriminant analysis with application to colorimetric sensor array data. *Technometrics* **57**(4), 524–534. MR3425489

[51] Zhou, H., Li, L., and Zhu, H. (2013). Tensor regression with applications in neuroimaging data analysis. *Journal of American Statistical Association*, **108**(502), 540–552. MR3174640

Boxiang Wang
Department of Statistics and Actuarial Science
University of Iowa
Iowa City, IA 52242
USA
E-mail address: boxiang-wang@uiowa.edu

Le Zhou
Department of Mathematics
Hong Kong Baptist University
Kowloon Tong
Hong Kong SAR
E-mail address: lezhou@hkbu.edu.hk

Jian Yang
Yahoo! Research
Sunnyvale, CA 94089
USA
E-mail address: jianyang@yahooinc.com

Qing Mai
Department of Statistics
Florida State University
Tallahassee, FL 32306
USA
E-mail address: qmai@fsu.edu