Quadratic Discriminant Revisited

Wenbo Cao, Robert Haralick
Dept. of Computer Science, The Graduate Center, City University of New York, New York, NY 10011 USA
Email: wenbo.cao@gmail.com, haralick@aim.com

***Abstract*—In this study, we revisit quadratic discriminant analysis (*QDA*). For this purpose, we present a majorize-minimize (*MM*) optimization algorithm to estimate parameters for generative classifiers, of which conditional distributions are from the exponential family. Furthermore, we propose a block-coordinate descent algorithm to sequentially update parameters of QDA in each iteration of the MM algorithm; for each update, we apply a trust region method, of which each iteration has a simple closed form solution. Numerical experiments show that: when compared with conjugate gradient method, the new proposed method is faster in 9 of 10 benchmark data sets; when compared with other widely used quadratic classifiers in the literature, QDA trained with the proposed method is either the best or not statistically significantly different from the best ones in 8 of 10 benchmark data sets.**

I. Introduction

Quadratic discriminant analysis (*QDA*) is a classical generative probabilistic method for classification problems. Essentially speaking, conditional distributions of QDA are assumed to be multivariate Gaussian; posterior distributions are derived via Bayes theorem, and used to classify observations into different classes [1], [2]. Traditionally, parameters of QDA are estimated by maximizing the joint likelihood of observations and their associate class labels. Though computational efficient, this generative approach does not aim at reducing classification error, and is not robust to model mis-specification. For recent studies on QDA, we refer to [3] and references thereof.

Generative and discriminative learning are two commonly used approaches for estimating parameters of given classifiers. It is well understood that generative learning is more statistically efficient when models are well specified; discriminative learning can achieve better classification accuracy when models are mis-specified [4], [5]. In the hope of taking advantages of both generative and discriminative learning, several authors proposed hybrid generative/discriminative learning for classification methods [6]–[8].

Motivated by recent works on hybrid learning, we propose to estimate parameters of QDA by maximizing a convex combination of the joint log-likelihood and conditional log-likelihood of given observations and their class labels. Our main contributions of this study are: 1) we present a general iterative algorithm for parameter estimation when conditional distributions of classifiers are from the well known exponential family; and 2) we revisit QDA, and present a block coordinate descent algorithm for solving a specific optimization problem derived for QDA.

The remainder of the paper is organized as follows. In section II, we give a brief introduction of generative classifiers and commonly used approaches for parameter estimation. In section III, we present an iterative optimization algorithm for parameter estimation when conditional distribution are from the exponential family. Multivariate Gaussian is a specific distribution of the exponential family; we revisit QDA and present a specific block-coordinate descent algorithm for estimating parameters of QDA in section IV. We present experimental results in section V, and conclude the study with the summary of our work, and possible future directions in section VI.

Throughout this paper, we use the following conventions unless otherwise specified: (1) vectors are column vectors and are represented as bold small letters, e.g. $\mathbf{x}$; (2) matrices are represented as bold capital letters, e.g. $\mathbf{X}$; (3) $\mathbf{I}$ is an identity matrix; (4) $\square^T$ is the transpose of a vector or matrix $\square$; (5) $\det(\mathbf{X})$ is the determinant of $\mathbf{X}$; (6) $\mathbf{X}^{-1}$ is the inverse of $\mathbf{X}$; $(7)\mathbf{X} \succeq \mathbf{Y}$ means that matrix $\mathbf{X} - \mathbf{Y}$ is positive semidefinite; and (8) we define the inner product of matrices $\mathbf{X} \in \mathbb{R}^{m\times n}$ and $\mathbf{Y} \in \mathbb{R}^{m\times n}$ as

$$\langle \mathbf{X}, \mathbf{Y} \rangle = \sum_{i=1}^{m}\sum_{j=1}^{n} X_{ij}Y_{ij}.$$

II. Generative Models for Classification

We consider a generative probabilistic model for classification problems. For this purpose, we assume, for the k-th class, a parametric density, $p(\mathbf{x}|k;\boldsymbol{\Theta}_k)$, which is described by a set of parameters $\boldsymbol{\Theta}_k$; here $k = 1, 2, \cdots, K$. We intend to keep the densities abstract at this moment, and will specify them in section III. Define prior probabilities as follows

$$p(k) = \frac{e^{\upsilon_k}}{\sum_{m=1}^{K} e^{\upsilon_m}}. \tag{1}$$

Posterior probabilities can be derived via Bayes theorem:

$$p(k|\mathbf{x};\boldsymbol{\Theta},\boldsymbol{\Upsilon}) = \frac{p(\mathbf{x}|k;\boldsymbol{\Theta}_k)e^{\upsilon_k}}{\sum_{k=1}^{K} p(\mathbf{x}|k;\boldsymbol{\Theta}_k)e^{\upsilon_k}}, \tag{2}$$

where $\boldsymbol{\Theta} = (\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \cdots, \boldsymbol{\Theta}_K)$ and $\boldsymbol{\Upsilon} = (\upsilon_1, \upsilon_2, \cdots, \upsilon_K)$ are unknown parameters. To minimize classification loss for 0-1 cost, we would assign an observation to a class which has the highest posterior probability [1]. Formally speaking, the decision rule to classify $\mathbf{x}$ is as follows

$$\hat{y} = \arg\max_k p(k|\mathbf{x};\boldsymbol{\Theta},\boldsymbol{\Upsilon}). \tag{3}$$

Consider a given training data set

$$\mathcal{D} = \{(\mathbf{x}_n, y_n)|n = 1, 2, \cdots, N, \mathbf{x}_n \in \mathbb{R}^d, y_n \in \{1, 2, \cdots, K\}\},$$

where $\mathbf{x}_n$ is the n-th observation, and y_n is its class label. Define index set $\mathcal{C}_k$ that contains the indices of all observations of class k in $\mathcal{D}$; that is,

$$\mathcal{C}_k = \{n|y_n = k, n = 1, 2, \cdots, N\}, \tag{4}$$

where $k = 1, 2, \cdots, K$. Let N_k be the number of observations of class k in the data set $\mathcal{D}$.

Our main focus is how to estimate the parameters of a generative model, $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$, using observations in the data set $\mathcal{D}$. Let $\mathcal{L}_G$ and $\mathcal{L}_D$ be the scaled negative joint log-likelihood and the scaled negative conditional log-likelihood, which are defined as follows

$$\begin{aligned}\mathcal{L}_G(\mathbf{\Theta},\mathbf{\Upsilon}) &= -\frac{1}{N}\sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k}\log[p(\mathbf{x}_n|k;\mathbf{\Theta}_k)\frac{e^{\upsilon_k}}{\sum_{m=1}^{K}e^{\upsilon_m}}]\\ \mathcal{L}_D(\mathbf{\Theta},\mathbf{\Upsilon}) &= -\frac{1}{N}\sum_{n=1}^{N}\log p(y_n|\mathbf{x}_n;\mathbf{\Theta},\mathbf{\Upsilon}).\end{aligned}$$

One approach for estimating $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$ is to maximize $\mathcal{L}_G$. Alternatively, $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$ can be estimated by maximizing $\mathcal{L}_D$.

Motivated by the recent works on hybrid generative/discriminative learning [6]–[8], we minimize the following objective function for estimating $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$ in this study.

$$\mathcal{L}(\mathbf{\Theta},\mathbf{\Upsilon}) = \beta\mathcal{L}_D(\mathbf{\Theta},\mathbf{\Upsilon}) + (1-\beta)\mathcal{L}_G(\mathbf{\Theta},\mathbf{\Upsilon}), \tag{5}$$

where $0 \le \beta \le 1$. As β varies from 0 to 1, we can interpolate between the two approaches mentioned above.

III. Learning For Exponential Family

In this study, we assume conditional distributions are from the exponential family. That is,

$$p(\mathbf{x}|k;\mathbf{\Theta}_k) = \exp\{h_k(\mathbf{x}) + \langle\mathbf{\Theta}_k, T_k(\mathbf{x})\rangle - A_k(\mathbf{\Theta}_k)\}, \tag{6}$$

where $T_k(\mathbf{x})$ is the potential function or sufficient statistics; $h_k(\mathbf{x})$ is a function that does not depend on $\mathbf{\Theta}_k$; $A_k(\mathbf{\Theta}_k)$ is the log partition function. For details about the exponential family, we refer to [9]. Canonical parameter $\mathbf{\Theta}_k \in \mathcal{P}_k$ needs to be estimated; $\mathcal{P}_k$ is the domain of $\mathbf{\Theta}_k$, which we assume to be convex in this study. We intend to keep $h_k(\mathbf{x})$'s and $T_k(\mathbf{x})$'s, $A_k(\mathbf{\Theta}_k)$'s and $\mathcal{P}_k$'s abstract at this moment, and will specify them when needed.

Following the definition in equation (5), we can write the objective function to be minimized as follows

$$\begin{aligned}\mathcal{L}(\mathbf{\Theta},\mathbf{\Upsilon}) &= -\frac{1}{N}\sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k}\log[p(\mathbf{x}_n|k;\mathbf{\Theta}_k)\frac{e^{\upsilon_k}}{\sum_{m=1}^{K}e^{\upsilon_m}}] \\ &\quad +\frac{\beta}{N}\sum_{n=1}^{N}\log\sum_{k=1}^{K}[e^{\upsilon_k}p(\mathbf{x}_n|k;\mathbf{\Theta}_k)\frac{e^{\upsilon_k}}{\sum_{m=1}^{K}e^{\upsilon_m}}].\end{aligned} \tag{7}$$

The optimization problem for estimating parameters $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$ can be written as

$$\begin{aligned}\min_{\mathbf{\Theta},\mathbf{\Upsilon}}\quad & \mathcal{L}(\mathbf{\Theta},\mathbf{\Upsilon}) \\ \text{s.t.}\quad & \mathbf{\Theta}_k \in \mathcal{P}_k \quad k=1,\cdots,K\end{aligned} \tag{8}$$

We note that the optimization problem is not convex, Thus we will be satisfied to find local optimums. In the following, we shall present a majorize-minimize (*MM*) algorithm in which we iteratively minimize a convex upper bound of the original objective function. For an introduction of the MM optimization method, we refer to [10] and references therein.

We note that, due to log-partition functions $A_k(\mathbf{\Theta}_k)$'s, the log-sum terms of $\mathcal{L}$ are generally not convex. Several lower bounds of log-partition functions have been proposed in the literature, for example, [11], [12]. As global bounds, they are complex and loose. Note that one important property of log-partition functions $A_k(\mathbf{\Theta}_k)$, $(k=1,2,\cdots,K)$, is that they are convex functions of $\mathbf{\Theta}_k$. Therefore we have

$$A_k(\mathbf{\Theta}_k) \ge A_k(\tilde{\mathbf{\Theta}}_k) + \left\langle \frac{\partial A_k}{\partial \mathbf{\Theta}_k}|_{\tilde{\mathbf{\Theta}}_k}, \mathbf{\Theta}_k - \tilde{\mathbf{\Theta}}_k \right\rangle. \tag{9}$$

Define an auxiliary function for $p(\mathbf{x}|k;\mathbf{\Theta}_k)$ at $\tilde{\mathbf{\Theta}}_k$ as follows,

$$q(\mathbf{x}|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k) = e^{h_k(\mathbf{x})+\langle\mathbf{\Theta}_k,T_k(\mathbf{x})\rangle - A_k(\tilde{\mathbf{\Theta}}_k) - \left\langle \frac{\partial A_k}{\partial\tilde{\mathbf{\Theta}}_k}|_{\tilde{\mathbf{\Theta}}_k}, \mathbf{\Theta}_k-\tilde{\mathbf{\Theta}}_k\right\rangle}. \tag{10}$$

Consequently, we have $p(\mathbf{x}|k;\mathbf{\Theta}_k) \le q(\mathbf{x}|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k)$. When $\mathbf{\Theta}_k = \tilde{\mathbf{\Theta}}_k$, $p(\mathbf{x}|k;\mathbf{\Theta}_k) = q(\mathbf{x}|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k)$. Define

$$\begin{aligned}\overline{\mathcal{L}}(\mathbf{\Theta},\mathbf{\Upsilon};\tilde{\mathbf{\Theta}}) &= \frac{1}{N}\sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k}\log[p(\mathbf{x}_n|k;\mathbf{\Theta}_k)\frac{e^{\upsilon_k}}{\sum_{m=1}^{K}e^{\upsilon_m}}] \\ &\quad -\frac{\beta}{N}\sum_{n=1}\log\sum_{k=1}^{K}[q(\mathbf{x}_n|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k)\frac{e^{\upsilon_k}}{\sum_{m=1}^{K}e^{\upsilon_m}}]\end{aligned} \tag{11}$$

Then we have $\overline{\mathcal{L}}(\mathbf{\Theta},\mathbf{\Upsilon};\tilde{\mathbf{\Theta}})$ *majorizes* $\mathcal{L}(\mathbf{\Theta},\mathbf{\Upsilon})$ at $\tilde{\mathbf{\Theta}}$: $\mathcal{L}(\mathbf{\Theta},\mathbf{\Upsilon}) \ge \overline{\mathcal{L}}(\mathbf{\Theta},\mathbf{\Upsilon};\tilde{\mathbf{\Theta}})$, and $\mathcal{L}(\tilde{\mathbf{\Theta}},\mathbf{\Upsilon}) = \overline{\mathcal{L}}(\tilde{\mathbf{\Theta}},\mathbf{\Upsilon};\tilde{\mathbf{\Theta}})$. Moreover, $\overline{\mathcal{L}}$ is a convex function.

Thus we have the following convex optimization problem

$$\begin{aligned}\min_{\mathbf{\Theta},\mathbf{\Upsilon}}\quad & \overline{\mathcal{L}}(\mathbf{\Theta},\mathbf{\Upsilon};\tilde{\mathbf{\Theta}}) \\ \text{s.t.}\quad & \mathbf{\Theta}_k \in \mathcal{P}_k, \quad k=1,\cdots,K\end{aligned} \tag{12}$$

Let $\mathbf{\Theta}^\star$ and $\mathbf{\Upsilon}^\star$ be optimal solutions of optimization problem (12). We note that, for any $\tilde{\mathbf{\Theta}}$ and $\tilde{\mathbf{\Upsilon}}$,

$$\mathcal{L}(\mathbf{\Theta}^\star,\mathbf{\Upsilon}^\star) \le \overline{\mathcal{L}}(\mathbf{\Theta}^\star,\mathbf{\Upsilon}^\star;\tilde{\mathbf{\Theta}}) \le \overline{\mathcal{L}}(\tilde{\mathbf{\Theta}},\tilde{\mathbf{\Upsilon}};\tilde{\mathbf{\Theta}}) = \mathcal{L}(\tilde{\mathbf{\Theta}},\tilde{\mathbf{\Upsilon}}).$$

That is $\mathcal{L}(\mathbf{\Theta}^\star,\mathbf{\Upsilon}^\star)$ is less than or equal to $\mathcal{L}(\tilde{\mathbf{\Theta}},\tilde{\mathbf{\Upsilon}})$. In principal, we can iteratively find a new point $(\mathbf{\Theta},\mathbf{\Upsilon})$ to reduce the function value of $\mathcal{L}$ until a local optimum is reached. We present in algorithm 1 the pseudo-code of the MM algorithm in which the hybrid objective function for the exponential family is minimized. We note that, instead of finding optimums of the optimization problem (12) in each iteration, we can find a feasible point that reduces the objective function.

1: $\tau \leftarrow 0$ {τ:iteartion count}
2: Initialize feasible $\mathbf{\Theta}^{(\tau)}$, and $\mathbf{\Upsilon}^{(\tau)}$ for problem (8)
3: **while** not converge **do**
4: $\quad\tilde{\mathbf{\Theta}} \leftarrow \mathbf{\Theta}^{(\tau)}$
5: $\quad$Obtain $(\mathbf{\Theta}^{(\tau+1)}, \mathbf{\Upsilon}^{(\tau+1)})$ by solving convex optimization problem (12);
6: $\quad\tau \leftarrow \tau + 1$
7: **end while**

Fig. 1. A MM optimization algorithm for estimating parameters of the exponential family

IV. QDA Revisited

In this section, we shall investigate how to apply algorithm 1 for estimation parameters of QDA. Define

$$\mathbf{\Theta}_k = (\mathbf{\Omega}_k, \boldsymbol{\theta}_k) = (\mathbf{\Sigma}_k^{-1}, \mathbf{\Sigma}_k^{-1}\boldsymbol{\mu}_k), \tag{13}$$

where $\mathbf{\Sigma}_k$ and $\boldsymbol{\mu}_k$ are, respectively, the covariance matrix and the mean for the $k-$th conditional distribution. In the following discussion, we shall use $\mathbf{\Omega}_k$, $\boldsymbol{\theta}_k$ and $\mathbf{\Sigma}_k$, $\boldsymbol{\mu}_k$ interchangeably. In QDA, conditional distributions are specified as multivariate Gaussian; thus for $k=1,2,\cdots,K$, we have the densities of conditional distribution expressed in the standard form of the exponential family

$$p(\mathbf{x}|k;\mathbf{\Theta}_k) = e^{h_k(\mathbf{x})+\langle\mathbf{\Theta}_k,T_k(\mathbf{x})\rangle - A_k(\mathbf{\Theta})},$$

where

$$h_k(\mathbf{x}) = -\frac{d}{2}\log(2\pi), \quad T_k(\mathbf{x}) = \left(-\tfrac{1}{2}\mathbf{x}\mathbf{x}^T, \quad \mathbf{x}\right), \tag{14}$$

$$A_k(\mathbf{\Theta}_k) = \frac{1}{2}\boldsymbol{\theta}_k^T\mathbf{\Omega}_k^{-1}\boldsymbol{\theta}_k - \frac{1}{2}\log\det\mathbf{\Omega}_k. \quad (15)$$

Note that

$$\frac{\partial}{\partial\mathbf{\Theta}_k}A_k(\mathbf{\Theta}_k) = \left(-\tfrac{1}{2}\mathbf{\Omega}_k^{-1}\boldsymbol{\theta}_k\boldsymbol{\theta}_k^T\mathbf{\Omega}_k^{-1} - \tfrac{1}{2}\mathbf{\Omega}_k^{-1}, \quad \boldsymbol{\mu}_k\right).$$

We thus have the auxiliary function for $p(\mathbf{x}|k;\mathbf{\Theta}_k)$ at $\tilde{\mathbf{\Theta}}_k = (\tilde{\mathbf{\Omega}}_k, \tilde{\boldsymbol{\theta}}_k)$ as following,

$$q(\mathbf{x}|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k) = e^{\frac{1}{2}\langle\tilde{\mathbf{M}}_k - \mathbf{x}\mathbf{x}^T, \mathbf{\Omega}_k\rangle + \langle\mathbf{x}-\tilde{\boldsymbol{\mu}}_k, \boldsymbol{\theta}_k\rangle + c_k}, \quad (16)$$

where

$$\begin{aligned}\tilde{\mathbf{M}}_k &= \tilde{\mathbf{\Omega}}_k^{-1} + \tilde{\mathbf{\Omega}}_k^{-1}\tilde{\boldsymbol{\theta}}_k\tilde{\boldsymbol{\theta}}_k^T\tilde{\mathbf{\Omega}}_k^{-1}\\ c_k &= -\frac{d}{2}\log(2\pi) - A_k(\tilde{\mathbf{\Theta}}_k) + \left\langle\tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\theta}_k}\right\rangle - \frac{1}{2}\left\langle\tilde{\mathbf{M}}_k, \tilde{\mathbf{\Omega}}_k\right\rangle.\end{aligned}$$

The convex upper bound of $\mathcal{L}(\mathbf{\Theta}, \mathbf{\Upsilon})$ is as follows

$$\begin{aligned}\overline{\mathcal{L}}(\mathbf{\Theta}, \mathbf{\Upsilon}; \tilde{\mathbf{\Theta}}) = \quad & \frac{\beta}{N}\sum_{n=1}^{N}\log[\sum_{k=1}^{K}e^{\upsilon_k}q(\mathbf{x}_n|k;\mathbf{\Theta}_k,\tilde{\mathbf{\Theta}}_k)] + (1-\beta)\log\sum_{k=1}^{K}e^{\upsilon_k}\\ & -\frac{1}{N}\sum_{k=1}^{K}\sum_{n\in\mathcal{C}_k}[\langle\mathbf{x}_n,\boldsymbol{\theta}_k\rangle - \frac{1}{2}\left\langle\mathbf{x}_n\mathbf{x}_n^T, \mathbf{\Omega}_k\right\rangle]\\ & -\sum_{k=1}^{K}\frac{N_k}{N}[\upsilon_k - \frac{1}{2}\boldsymbol{\theta}_k^T\mathbf{\Omega}_k^{-1}\boldsymbol{\theta}_k + \frac{1}{2}\log\det\mathbf{\Omega}_k].\end{aligned} \quad (17)$$

We require covariance matrices being positive definite; i.e. $\mathbf{\Sigma}_k \succ \mathbf{0}$. Equivalently, we can have $\mathbf{\Omega}_k \succ \mathbf{0}$. In practice, we can have $\mathbf{\Omega}_k \succeq \alpha\mathbf{I}$, where $\alpha > 0$.

We can apply algorithm 1 to estimate $\mathbf{\Theta}$ and $\mathbf{\Upsilon}$. In each iteration of algorithm 1, we need solve the following convex optimization problem

$$\begin{aligned}\min_{\mathbf{\Theta},\mathbf{\Upsilon}} \quad & \overline{\mathcal{L}}(\mathbf{\Theta}, \mathbf{\Upsilon}; \tilde{\mathbf{\Theta}})\\ \text{s.t.} \quad & \mathbf{\Omega}_k \succeq \alpha\mathbf{I}, \quad k = 1, 2, \cdots, K\end{aligned} \quad (18)$$

where $\alpha > 0$ is given. To the best of our knowledge, there is no existing optimization method to solve this problem. In the following, we will present a block-coordinate descent algorithm for problem (18) that sequentially updates $\mathbf{\Upsilon}$, $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, …, $\boldsymbol{\theta}_K$ and $\mathbf{\Omega}_1$, $\mathbf{\Omega}_2$, …, $\mathbf{\Omega}_k$. We will discuss the details of optimization algorithms for $\mathbf{\Upsilon}$ in subsection IV-A, for $\boldsymbol{\theta}_k$ in subsection IV-B, and for $\mathbf{\Omega}_k$ in subsection IV-C; we will discuss selecting initial points, early stopping and β in subsection IV-D.

A. Updating υ_k's

We assume $\mathbf{\Theta}_k$'s are fixed: for $k = 1, 2, \cdots, K$, $\boldsymbol{\theta}_k = \tilde{\boldsymbol{\theta}}_k$, and $\mathbf{\Omega}_k = \tilde{\mathbf{\Omega}}_k$. Since $q(\mathbf{x}_n|k;\tilde{\mathbf{\Theta}}_k,\tilde{\mathbf{\Theta}}_k) = p(\mathbf{x}_n|k;\tilde{\mathbf{\Theta}}_k)$, we have the optimization problem for $\mathbf{\Upsilon} = (\upsilon_1, \upsilon_2, \cdots, \upsilon_K)$ as follows

$$\min_{\mathbf{\Upsilon}} -\sum_{k=1}^{K}\frac{N_k}{N}\upsilon_k + \frac{\beta}{N}\sum_{n=1}^{N}\log\sum_{k=1}^{K}p(\mathbf{x}_n|k;\tilde{\mathbf{\Theta}}_k)e^{\upsilon_k} + (1-\beta)\log\sum_{k=1}^{K}e^{\upsilon_k}.$$

B. Updating θ_k's

In this subsection, we consider υ_k's and $\mathbf{\Omega}_k$'s are fixed; that is, for $k = 1, \cdots, K$, $\upsilon_k = \upsilon_k^\star$, and $\mathbf{\Omega}_k = \mathbf{\Omega}_k$. Furthermore, we consider all $\boldsymbol{\theta}_k$'s but $\boldsymbol{\theta}_m$ are fixed; that is $\forall k < m, \boldsymbol{\theta}_k = \boldsymbol{\theta}_k^\star$; and $\forall k > m$, $\boldsymbol{\theta}_k = \tilde{\boldsymbol{\theta}}_k$. We define $f(\boldsymbol{\theta}_m)$ in this subsection as follows,

$$f(\boldsymbol{\theta}_m) = \overline{\mathcal{L}}(\boldsymbol{\theta}_1^\star, \cdots, \boldsymbol{\theta}_{m-1}^\star, \boldsymbol{\theta}_m, \tilde{\boldsymbol{\theta}}_{m+1}, \cdots, \tilde{\boldsymbol{\theta}}_K, \tilde{\mathbf{\Omega}}_1, \cdots, \tilde{\mathbf{\Omega}}_K, \mathbf{\Upsilon}^\star). \quad (19)$$

In the following, we will present a trust region method to minimize $f(\boldsymbol{\theta}_m)$. Due to the space limit, we refer to [13], [14] for details about trust region method.

Define

$$\boldsymbol{\phi}_m = \boldsymbol{\theta}_m - \boldsymbol{\theta}_m^{(\tau)}, \quad (20)$$

and

$$q^{(\tau)}(m|\mathbf{x}_n) = \frac{e^{\upsilon_m^\star}q(\mathbf{x}_n|m;\mathbf{\Theta}_m^{(\tau)},\tilde{\mathbf{\Theta}}_m)}{\sum_{k=1}^{K}e^{\upsilon_k^\star}q(\mathbf{x}_n|k;\mathbf{\Theta}_k^{(\tau)},\tilde{\mathbf{\Theta}}_k)}$$

where $\boldsymbol{\theta}_m^{(\tau)}$ is an estimation of $\boldsymbol{\theta}_m$ at τ-th iteration, and

$$\mathbf{\Theta}_k^{(\tau)} = \begin{cases}(\tilde{\mathbf{\Omega}}_k, \boldsymbol{\theta}_k^\star), & k = 1, 2, \cdots, m-1\\ (\tilde{\mathbf{\Omega}}_k, \boldsymbol{\theta}_k^{(\tau)}), & k = m\\ (\tilde{\mathbf{\Omega}}_k, \tilde{\boldsymbol{\theta}}_k), & k = m+1, m+2, \cdots, K.\end{cases}$$

We need solve the following trust region subproblem in the τ-th iteration,

$$\begin{aligned}\min_{\boldsymbol{\phi}_m} \quad & \tfrac{\lambda}{2}\boldsymbol{\phi}_m^T\boldsymbol{\phi}_m + \mathbf{g}_m^T\boldsymbol{\phi}_m\\ \text{s.t.} \quad & \boldsymbol{\phi}_m^T\boldsymbol{\phi}_m \le \epsilon^{(\tau)},\end{aligned} \quad (21)$$

where

$$\mathbf{g}_m = \frac{1}{N}\{\sum_{n=1}^{N}\beta q^{(\tau)}(m|\mathbf{x}_n)(\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_m) - \sum_{n\in\mathcal{C}_m}(\mathbf{x}_n - \boldsymbol{\mu}_m^{(\tau)})\},$$

and λ is the $1-$norm of the Hessian matrix $\frac{\beta}{N}\sum_{n=1}^{N}q^{(\tau)}(m|\mathbf{x}_n)[1-q^{(\tau)}(m|\mathbf{x}_n)](\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_m)(\mathbf{x}_n - \tilde{\boldsymbol{\mu}}_m)^T + \frac{N_m}{N}\tilde{\mathbf{\Omega}}_m^{-1}$. It can be shown that the optimal solution of problem (21) is

$$\boldsymbol{\phi}_m^* = -\frac{\mathbf{g}_m}{\max(\sqrt{\frac{\mathbf{g}_m^T\mathbf{g}_m}{\epsilon^{(\tau)}}}, \lambda)}. \quad (22)$$

To summarize, we present the pseudo code of the trust region algorithm for finding $\boldsymbol{\theta}_m^\star$ in algorithm 2.

Specify $\eta_0, \eta_1, \eta_2, \eta_3, \eta_4$ {see [14]}
$\tau \leftarrow 0$ {τ:iteartion count}
$\boldsymbol{\theta}_m^{(\tau)} \leftarrow \tilde{\boldsymbol{\theta}}_m$
while not converge **do**
  $\boldsymbol{\phi}_m^* = -\mathbf{g}_m/\max(\sqrt{\frac{\mathbf{g}_m^T\mathbf{g}_m}{\epsilon^{(\tau)}}}, \lambda)$
  $r = \frac{f(\boldsymbol{\theta}_m+\boldsymbol{\phi}_m^*)-f(\boldsymbol{\theta}_m)}{\mathbf{g}_m^T\boldsymbol{\phi}_m^* + 0.5\lambda(\boldsymbol{\phi}_m^*)^T\boldsymbol{\phi}_m^*}$
  if $r > \eta_0$, $\boldsymbol{\theta}_m^{(\tau+1)} \leftarrow \boldsymbol{\theta}_m^{(\tau)} + \boldsymbol{\phi}_m^*$; otherwise, $\boldsymbol{\theta}_m^{(\tau+1)} \leftarrow \boldsymbol{\theta}_m^{(\tau)}$
  if $r \ge \eta_2$, update $\epsilon^{(\tau+1)} \in [\epsilon^{(\tau)}, \eta_1\epsilon^{(\tau)}]$; otherwise, update $\epsilon^{(\tau+1)} \in [\eta_3\|\boldsymbol{\phi}_m^*\|_2, \eta_4\epsilon^{(\tau)}]$;
  $\tau \leftarrow \tau + 1$
end while

Fig. 2. A trust-region algorithm for finding $\boldsymbol{\theta}_m^\star$

C. Updating $\mathbf{\Omega}_k$'s

In this subsection, we consider υ_k's and $\boldsymbol{\theta}_k$'s are fixed; that is, for $k = 1, \cdots, K$, $\upsilon_k = \upsilon_k^\star$, and $\boldsymbol{\theta}_k = \boldsymbol{\theta}_k^\star$. Furthermore, consider all $\mathbf{\Omega}_k$'s but $\mathbf{\Omega}_m$ are fixed; that is $\forall k < m$, $\mathbf{\Omega}_k = \mathbf{\Omega}_k^\star$; $\forall k > m$, $\mathbf{\Omega}_k = \tilde{\mathbf{\Omega}}_k$. We define $h(\mathbf{\Omega}_m)$ in this subsection as follows,

$$h(\mathbf{\Omega}_m) = \overline{\mathcal{L}}(\boldsymbol{\theta}_1^\star, \cdots, \boldsymbol{\theta}_K^\star, \mathbf{\Omega}_1^\star, \cdots, \mathbf{\Omega}_{m-1}^\star, \mathbf{\Omega}_m, \tilde{\mathbf{\Omega}}_{m+1}, \cdots, \tilde{\mathbf{\Omega}}_K, \boldsymbol{\Upsilon}^\star). \tag{23}$$

So the optimization problem for $\mathbf{\Omega}_m$ is

$$\begin{aligned} \min_{\mathbf{\Omega}_m} \quad & h(\mathbf{\Omega}_m) \\ \text{s.t.} \quad & \mathbf{\Omega}_m \succeq \alpha \mathbf{I}. \end{aligned}$$

To our best knowledge, there is no existing method to solve this optimization problem. We will present an iterative trust region method to solve this optimization problem in the remaining of this subsection.

Let $\mathbf{\Omega}_m^{(\tau)}$ be an estimation of $\mathbf{\Omega}_m$ at τ-th iteration. We define

$$\mathbf{\Phi}_m = (\mathbf{\Omega}_m^{(\tau)})^{-\frac{1}{2}} (\mathbf{\Omega}_m - \mathbf{\Omega}_m^{(\tau)}) (\mathbf{\Omega}_m^{(\tau)})^{-\frac{1}{2}}. \tag{24}$$

Then by $\mathbf{\Omega}_m \succeq \alpha \mathbf{I}$, we have

$$\mathbf{\Phi}_m \succeq \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I}.$$

Define

$$q^{(\tau)}(m|\mathbf{x}_n) = \frac{e^{\upsilon_m^\star} q(\mathbf{x}_n | m; \mathbf{\Theta}_m^{(\tau)}, \tilde{\mathbf{\Theta}}_m)}{\sum_{k=1}^K e^{\upsilon_k^\star} q(\mathbf{x}_n | k; \mathbf{\Theta}_k^{(\tau)}, \tilde{\mathbf{\Theta}}_k)}$$

where

$$\mathbf{\Theta}_k^{(\tau)} = \begin{cases} (\mathbf{\Omega}_k^\star, \boldsymbol{\theta}_k^\star), & k = 1, 2, \cdots, m \\ (\mathbf{\Omega}_k^{(\tau)}, \boldsymbol{\theta}_k^\star), & k = m \\ (\tilde{\mathbf{\Omega}}_k, \boldsymbol{\theta}_k^\star), & k = m+1, m+2, \cdots, K, \end{cases}$$

We need solve the following trust region subproblem in the τ-th iteration

$$\begin{aligned} \min_{\mathbf{\Phi}_m} \quad & \tfrac{\lambda}{2} \langle \mathbf{\Phi}_m, \mathbf{\Phi}_m \rangle + \langle \mathbf{G}_m, \mathbf{\Phi}_m \rangle \\ \text{s.t.} \quad & \mathbf{\Phi}_m \succeq \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \\ & \langle \mathbf{\Phi}_m, \mathbf{\Phi}_m \rangle \le \epsilon^{(\tau)}, \end{aligned}$$

where

$$\lambda = \frac{N_m}{N} \{ (\boldsymbol{\theta}_m^\star)^T (\mathbf{\Omega}_m^{(\tau)})^{-1} \boldsymbol{\theta}_m^\star + \frac{1}{2} \}, \tag{25}$$

and

$$\mathbf{G}_m = \frac{1}{2N} \sqrt{\mathbf{\Omega}_m^{(\tau)}} \{ \beta \sum_{n=1}^N q^{(\tau)}(m|\mathbf{x}_n) [\tilde{\mathbf{M}}_m - \mathbf{x}_n \mathbf{x}_n^T] + \sum_{n \in \mathcal{C}_m} \mathbf{S}_n^{(\tau)} \} \sqrt{\mathbf{\Omega}_m^{(\tau)}}, \tag{26}$$

where $\mathbf{S}_n^{(\tau)} = (\mathbf{\Omega}_m^{(\tau)})^{-1} + (\mathbf{\Omega}_m^{(\tau)})^{-1} \boldsymbol{\theta}_m^\star (\boldsymbol{\theta}_m^\star)^T (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{x}_n \mathbf{x}_n^T$. In appendix A, we show that we can approximately solve this problem by [1]

$$\mathbf{\Phi}_m^* = \sqrt{\min(\frac{\epsilon^{(\tau)}}{\rho}, 1)} \{ \mathbf{C}_-^{(\tau)} + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \}, \tag{27}$$

where

$$\begin{aligned} \mathbf{C}^{(\tau)} &= \frac{1}{\lambda} \mathbf{G}_m + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \\ \rho &= \| \mathbf{C}_-^{(\tau)} + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \|_F^2, \end{aligned}$$

and $\mathbf{C}_-^{(\tau)}$ is the negative semidefinite parts of $\mathbf{C}^{(\tau)}$; let $\mathbf{C}^{(\tau)} = \sum_{i=1}^d \lambda_i \mathbf{e}_i \mathbf{e}_i^T$ be the eigen-decomposition of $\mathbf{C}^{(\tau)}$, then $\mathbf{C}_-^{(\tau)} = \sum_{i=1}^d \max(-\lambda_i, 0) \mathbf{e}_i \mathbf{e}_i^T$, (see [15]).

In summary, we present in Algorithm 3 the pseudo-code of the trust region algorithm for finding $\mathbf{\Omega}_m^\star$.

Specify $\eta_0, \eta_1, \eta_2, \eta_3, \eta_4$ {see [14]}
$\tau \leftarrow 0$ {τ:iteartion count}
$\mathbf{\Omega}_m^{(\tau)} \leftarrow \tilde{\mathbf{\Omega}}_m$
while not converge **do**
  calculate $\mathbf{\Phi}_m^*$ as described in equation (27)
  $\hat{\mathbf{\Omega}}_m = \mathbf{\Omega}_m^{(\tau)} + (\mathbf{\Omega}_m^{(\tau)})^{1/2} \mathbf{\Phi}_m^* (\mathbf{\Omega}_m^{(\tau)})^{1/2}$
  $r = \frac{h(\hat{\mathbf{\Omega}}_m) - h(\mathbf{\Omega}_m)}{0.5\lambda \langle \mathbf{\Phi}_m^*, \mathbf{\Phi}_m^* \rangle + \langle \mathbf{G}_m, \mathbf{\Phi}_m^* \rangle}$
  if $r > \eta_0$, $\mathbf{\Omega}_m^{(\tau+1)} \leftarrow \hat{\mathbf{\Omega}}_m$; otherwise, $\mathbf{\Omega}_m^{(\tau+1)} \leftarrow \mathbf{\Omega}_m^{(\tau)}$
  if $r \ge \eta_2$, update $\epsilon^{(\tau+1)} \in [\epsilon^{(\tau)}, \eta_1 \epsilon^{(\tau)}]$; otherwise, update $\epsilon^{(\tau+1)} \in [\eta_3 \|\boldsymbol{\phi}_m^*\|_2, \eta_4 \epsilon^{(\tau)}]$;
  $\tau \leftarrow \tau + 1$
end while

Fig. 3. A trust-region algorithm for finding $\mathbf{\Omega}_m^\star$

D. Initial Points, Early Stopping and β

We use the following as our initial points,

$$\mathbf{\Omega}_k^{(0)} = \text{diag}(\hat{\mathbf{\Sigma}}_k)^+ + \alpha \mathbf{I} \quad \boldsymbol{\theta}_k^{(0)} = \tfrac{1}{N_k} \textstyle\sum_{n \in \mathcal{C}_k} \mathbf{\Omega}_k^{(0)} \mathbf{x}_n \quad \boldsymbol{\upsilon}_k^{(0)} = \log \frac{N_k}{N}.$$

where $\text{diag}(\hat{\mathbf{\Sigma}}_k)$ is the diagonal sample covariance matrices for class k; $\square^+$ is the pseudo-inverse of a matrix $\square$.

Early stopping has been shown to be an effective regularization method for boosting algorithms [16], [17]. Motivated by their works, we use early stopping in algorithm 1. We use cross-validation to determine β and the early stopping step in the iterative MM algorithm.

V. Numerical Experiments

We downloaded 10 datasets from UCI machine learning repository [18]. The statistics of the selected data sets are listed in table I. In our numerical experiments, we followed the following procedure: for all datasets, we randomly selected 40% of the data as training sets, and the remaining data are used as test sets; we standardized training data using the sample mean and standard deviation of training data; all methods are trained with training sets, and evaluated on the corresponding test sets unless otherwise specified. We repeated this procedure 20 times for each dataset.

We examined the influence of β on classification performance by varying β from 0 (discriminative) to 1 (generative). For Datasets diabetes, heart, ionosphere and sonar, we showed box plots of test set error rates in figure 4. For diabetes and heart, discriminative learning ($\beta = 0$) gives the best performance; for ionosphere and sonar, combinations of discriminative and generative learning gave best performance.

Bouchard and Triggs [6] used conjugate gradient (*CG*) method to optimize the same objective function as ours. We compared the new proposed method against CG for training QDA. For this purpose, we disabled early stopping in our method, and performed numerical experiments on an Intel

[1] To be precise, $\mathbf{\Phi}_m^*$ is a solution of the following feasibility problem: $\frac{\lambda}{2} \langle \mathbf{\Phi}_m, \mathbf{\Phi}_m \rangle + \langle \mathbf{G}_m, \mathbf{\Phi}_m \rangle \le 0$, $\mathbf{\Phi}_m \succeq \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I}$, and $\langle \mathbf{\Phi}_m, \mathbf{\Phi}_m \rangle \le \epsilon$.

	d	N	K
Diabetes	8	768	2
Heart	13	270	2
Parkinsons	22	195	2
Ionosphere	33	351	2
Sonar	60	208	2
Thyroid	5	215	3
Vertebral Column	6	310	3
Wine	13	178	3
Splice	60	3175	3
Breast Tissue	9	106	6

TABLE I. DESCRIPTIONS OF 10 BENCHMARK DATASETS (K: NUMBER OF CLASSES, d: DIMENSION, N: NUMBER OF OBSERVATIONS)

	CG QDA		Our Method	
	CPU Time (sec.)	Obj. Value	CPU Time (sec.)	Obj. Value
Diabetes	0.19 ± 0.08	$\mathbf{5.70 \pm 0.05}$	$\mathbf{0.06 \pm 0.01}$	$\mathbf{5.70 \pm 0.05}$
Heart	0.18 ± 0.01	$\mathbf{8.32 \pm 0.10}$	$\mathbf{0.09 \pm 0.02}$	$\mathbf{8.32 \pm 0.10}$
Parkinsons	10.46 ± 1.12	-2.47 ± 0.60	$\mathbf{2.91 \pm 1.35}$	$\mathbf{-9.09 \pm 1.70}$
Ionosphere	10.99 ± 2.18	$\mathbf{4.88 \pm 1.03}$	$\mathbf{10.13 \pm 2.26}$	5.52 ± 1.01
Sonar	23.24 ± 2.04	-17.29 ± 4.80	$\mathbf{3.19 \pm 4.24}$	$\mathbf{-92.01 \pm 14.37}$
Thyroid	1.13 ± 0.33	$\mathbf{1.04 \pm 0.22}$	$\mathbf{0.40 \pm 0.23}$	1.07 ± 0.23
Vertebral	4.01 ± 1.49	1.54 ± 0.13	$\mathbf{0.39 \pm 0.11}$	$\mathbf{-0.41 \pm 0.08}$
Wine	1.73 ± 0.96	$\mathbf{4.67 \pm 0.24}$	$\mathbf{0.89 \pm 0.22}$	4.87 ± 0.25
Splice	$\mathbf{67.59 \pm 31.53}$	39.38 ± 0.34	84.06 ± 80.61	$\mathbf{39.35 \pm 0.70}$
Breast Tissue	10.89 ± 0.67	-3.86 ± 0.62	$\mathbf{6.44 \pm 0.92}$	$\mathbf{-9.53 \pm 2.54}$

TABLE II. CPU TIME (SECONDS) AND FINAL OBJECTIVE FUNCTION VALUES OF TRAINING SETS ($\beta = 0.5$): CG V.S. OUR METHOD

Xeon (3.30GHz) computer with 16GB RAM. CG method reported local optimums within 3000 iterations for all datasets; our methods reported local optimums within 3000 iterations for all datasets except ionosphere. Computational speed and objective function values for both methods are reported in table V. In 9 of 10 datasets, our method is faster than CG method; in 7 of 10 datasets, our method terminates with equal or better local optimums.

To evaluate the classification performance of the new proposed method, we compared it with some widely used quadratic classifiers in the literature: naive Bayes, QDA [2], regularized discriminant analysis (*RDA*) [19] [3],Bayesian quadratical discriminant analysis (*BDA7*) [3] [4], L2-regularized logistic regression with quadratic terms (*LR*), SVM with polynomial kernel ($d = 2$) (*QSVM*). For RDA, we used leave-one-out approach to select optimal parameters on a $(0, 0.1, \cdots, 1) \times (0, 0.1, \cdots, 1)$ grid. For BDA7, we used the same settings as described in [3], and used leave-one-out approach to select optimal parameter from 42 parameter choices. For our method, we let $\alpha = 10^{-6}$ and $\beta = (0, 0.1, \cdots, 1.0)$, and used 10 fold cross validation to select optimal β and early stopping iteration. For CG QDA [6], we used the optimal β selected from our methods, and trained QDA with $\frac{1}{2}$-, 1- and 2-times of iterations determined from our methods; we recorded the smallest test set error rates. For LR and QSVM, we searched regularization coefficients in $[0, 10]$ by 10-fold cross validation.

Mean error rates for 10 benchmark datasets are reported in table V. For Naive Bayes and QDA, we reported "$-$" when the matlab function classify failed. The smallest mean error rates are emphasized with bold numbers. We also reported other mean error rates in bold, which are not statistically significantly different from the best ones; for this purpose, we used Wilcoxon signed-rank test with significant level 0.05 to compare error rates. As shown in table V, in 8 of the 10 datasets our method is either the best one, or not statistically significantly different from the best ones. The experiments shows that our method is competitive to other widely used quadratic classifiers in the literature.

VI. CONCLUSION

Quadratic discriminant analysis is a standard tool for classification problems. Classical QDA sacrifices classification accuracy for computation efficiency, which is not desired for classification problems. Motivated by the recent studies in hybrid generative/discriminative learning, we argue that, in order to obtain better classification accuracy, parameters of QDA can be estimated by maximizing a convex combination joint log-likelihood and conditional log-likelihood of given observations and their labels. For this purpose, we presented a MM optimization algorithm to estimate parameters for generative classifiers, of which conditional distributions are from the well known exponential family. Furthermore, we proposed a block-coordinate descent algorithm to sequentially update parameters of QDA in each iteration of the MM algorithm. For each update, we used a trust region method, of which each iteration has a simple closed form solution. Our numerical experiments show that our method is competitive with other well-known quadratic classification methods in the literature.

Our method can be easily adapted for linear discriminant analysis (*LDA*). Moreover in spirit, our MM algorithm and block-coordinate descent algorithm can be applied for any generative classifiers of which conditional distributions are from the exponential family.

Sparse parameters are desired when handling small sample problems, e.g. [20], [21]. Though early stopping provides regularization in our algorithm, we think explicitly adding sparse constraint via L_1 or L_0 regularization in the block-coordinate descent algorithm might provide additional robustness for QDA.

Recently, there is a growing interest in semi-supervised learning; that is to use unlabelled data in training classification methods [22]. Part of our future work is to include unlabelled data and missing data for estimating parameters for LDA and QDA.

APPENDIX

Let us first consider the following relaxed optimization problem,

$$\begin{aligned} \min_{\mathbf{\Phi}_m} \quad & \tfrac{\lambda}{2} \langle \mathbf{\Phi}_m, \mathbf{\Phi}_m \rangle + \langle \mathbf{G}_m, \mathbf{\Phi}_m \rangle \\ \text{s.t.} \quad & \mathbf{\Phi}_m \succeq \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \end{aligned}$$

Following [15], [23], [24], we can show that optimal solution for the relaxed problem is

$$\hat{\mathbf{\Phi}}_m = \mathbf{C}_-^{(\tau)} + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I}$$

where $\mathbf{C}^{(\tau)} = \frac{1}{\lambda} \mathbf{G}_m + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I}$. Define

$$\rho = \| \mathbf{C}_-^{(\tau)} + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \|_F^2,$$

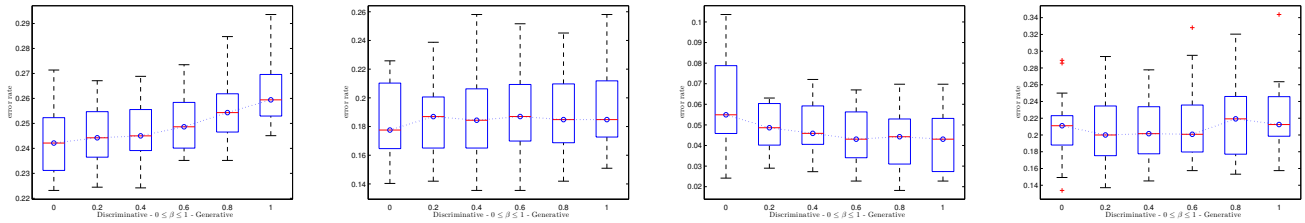
If $\rho \leq \epsilon^{(\tau)}$, then we can let $\mathbf{\Phi}_m^* = \hat{\mathbf{\Phi}}_m$; if $\rho > \epsilon^{(\tau)}$, then we can let $\mathbf{\Phi}_m^* = \sqrt{\frac{\epsilon^{(\tau)}}{\rho}} \hat{\mathbf{\Phi}}_m$. Therefore, we can let

$$\mathbf{\Phi}_m^* = \sqrt{\min(\frac{\epsilon^{(\tau)}}{\rho}, 1)} \{ \mathbf{C}_-^{(\tau)} + \alpha (\mathbf{\Omega}_m^{(\tau)})^{-1} - \mathbf{I} \},$$

[2] We use matlab function classify for naive Bayes and QDA

[3] We use code from **http://www-stat.stanford.edu/~jhf**

[4] We modified code from **http://www.ee.washington.edu/research/guptalab**.



(a) Diabetes (b) Heart (c) Ionosphere (d) Sonar

Fig. 4. Influence of coefficient of β on test set error rates: $\beta = 0$ is discriminative, and $\beta = 1$ is generative.

	Naive Bayes	QDA	RDA	BDA7	LR	QSVM	CG QDA	Our Method
Diabetes	25.61 ± 1.44	26.68 ± 1.88	$\mathbf{23.86 \pm 1.37}$	25.75 ± 1.21	$\mathbf{23.56 \pm 1.38}$	$\mathbf{23.34 \pm 1.34}$	24.39 ± 1.11	24.09 ± 1.06
Heart	17.44 ± 2.68	22.68 ± 3.39	$\mathbf{16.56 \pm 1.93}$	17.87 ± 2.31	18.52 ± 2.74	21.64 ± 3.38	20.38 ± 3.66	18.49 ± 2.53
Parkinsons	30.28 ± 3.04	–	17.67 ± 3.29	$\mathbf{12.96 \pm 3.18}$	16.41 ± 5.28	$\mathbf{14.24 \pm 1.90}$	$\mathbf{13.69 \pm 2.26}$	$\mathbf{14.68 \pm 4.07}$
Ionosphere	–	–	6.75 ± 1.60	7.45 ± 2.07	9.02 ± 1.92	14.40 ± 2.73	$\mathbf{5.64 \pm 1.89}$	$\mathbf{4.90 \pm 1.29}$
Sonar	32.25 ± 5.01	–	25.10 ± 4.47	22.65 ± 4.90	22.11 ± 3.11	22.58 ± 2.98	$\mathbf{20.75 \pm 3.59}$	$\mathbf{19.85 \pm 4.01}$
Thyroid	$\mathbf{3.92 \pm 1.00}$	5.65 ± 2.27	6.01 ± 2.16	4.87 ± 1.54	8.58 ± 2.93	7.86 ± 3.92	4.35 ± 0.97	$\mathbf{4.00 \pm 1.27}$
Vertebral Column	19.17 ± 2.80	18.35 ± 2.77	19.15 ± 3.72	$\mathbf{16.81 \pm 2.04}$	$\mathbf{16.90 \pm 2.48}$	$\mathbf{17.71 \pm 3.33}$	$\mathbf{17.22 \pm 2.42}$	$\mathbf{16.60 \pm 2.18}$
Wine	$\mathbf{3.76 \pm 1.88}$	–	$\mathbf{2.67 \pm 1.37}$	$\mathbf{3.75 \pm 2.56}$	3.88 ± 1.73	4.28 ± 2.18	$\mathbf{3.51 \pm 2.44}$	$\mathbf{2.85 \pm 1.62}$
Splice	–	–	11.06 ± 1.00	$\mathbf{7.21 \pm 1.86}$	10.88 ± 0.81	11.94 ± 0.75	8.93 ± 2.55	$\mathbf{8.16 \pm 3.22}$
Breast Tissue	$\mathbf{38.03 \pm 5.55}$	–	41.91 ± 5.95	$\mathbf{39.75 \pm 6.45}$	45.51 ± 5.86	48.04 ± 3.68	$\mathbf{37.00 \pm 5.41}$	$\mathbf{37.49 \pm 6.49}$

TABLE III. MEAN AND STANDARD DEVIATION OF TEST SET ERROR RATES FOR 10 DATASETS (SHOWN IN PERCENTAGE)

Since $\alpha(\mathbf{\Omega}_m^{(\tau)})^{-1} \preceq \mathbf{I}$, and $\min(\frac{\epsilon^{(\tau)}}{\rho}, 1) \leq 1$, we have $\sqrt{\min(\frac{\epsilon^{(\tau)}}{\rho}, 1)}[\alpha\mathbf{\Omega}_m^{-1} - \mathbf{I})] \succeq \alpha\mathbf{\Omega}_m^{-1} - \mathbf{I}$. Therefore, $\mathbf{\Phi}_m^* \succeq \alpha\mathbf{\Omega}_m^{-1} - \mathbf{I}$.

Note that when $\rho > \epsilon^{(\tau)}$, $\mathbf{\Phi}_m^*$ is not an optimal solution. Since $\rho > \epsilon^{(\tau)}$, we have

$$\frac{\lambda}{2}\langle \mathbf{\Phi}_m^*, \mathbf{\Phi}_m^* \rangle + \langle \mathbf{G}_m, \mathbf{\Phi}_m^* \rangle \leq \sqrt{\frac{\epsilon^{(\tau)}}{\rho}}(\frac{\lambda}{2}\left\langle \hat{\mathbf{\Phi}}_m, \hat{\mathbf{\Phi}}_m \right\rangle + \left\langle \mathbf{G}_m, \hat{\mathbf{\Phi}}_m \right\rangle) \leq 0.$$

That is $\mathbf{\Phi}_m^*$ does reduce the objective function value.

REFERENCES

[1] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*, 2nd ed. Wiley, Nov. 2000.

[2] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*, corrected ed. Springer, Jul. 2003.

[3] S. Srivastava, M. R. Gupta, and B. A. Frigyik, "Bayesian quadratic discriminant analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 1277–1305, Dec. 2007.

[4] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *NIPS*, 2001, pp. 841–848.

[5] P. Liang and M. I. Jordan, "An asymptotic analysis of generative, discriminative, and pseudolikelihood estimators," in *Proceedings of the 25th international conference on Machine learning*, ser. ICML '08, 2008, pp. 584–591.

[6] G. Bouchard and B. Triggs, "The trade-off between generative and discriminative classifiers," in *Proceedings in Computational Statistics, 16th Symposium of IASC*. Physica-Verlag, 2004, pp. 721–728.

[7] J. A. Lasserre, C. M. Bishop, and T. P. Minka, "Principled hybrids of generative and discriminative models," ser. CVPR '06, 2006, pp. 87–94.

[8] A. McCallum, C. Pal, G. Druck, and X. Wang, "Multi-conditional learning: generative/discriminative training for clustering and classification," ser. AAAI'06, 2006, pp. 433–439.

[9] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Found. Trends Mach. Learn.*, vol. 1, no. 1-2, pp. 1–305, Jan. 2008.

[10] D. R. Hunter and K. Lange, "A tutorial on mm algorithms," *Amer. Statist*, vol. 58, no. 1, pp. 30–37, 2004.

[11] M. Wainwright and M. Jordan, "Log-determinant relaxation for approximate inference in discrete markov random fields," *Trans. Sig. Proc.*, vol. 54, no. 6, pp. 2099–2109, Jun. 2006.

[12] T. Jebara and A. Pentland, "On reversing jensen's inequality," in *In Advances in Neural Information Processing Systems 13*. MIT Press, 2000, p. 2000.

[13] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-region methods*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000.

[14] Y. Yuan, "A review of trust region algorithms for optimization," in *ICIAM 99: Proceedings of the Fourth International Congress on Industrial & Applied Mathematics, Edinburgh*, 2000, pp. 271–282.

[15] S. Boyd and L. Xiao, "Least-squares covariance matrix adjustment," *SIAM J. Matrix Anal. Appl.*, vol. 27, no. 2, pp. 532–546, June 2005.

[16] T. Zhang and B. Yu, "Boosting with early stopping: Convergence and consistency," *Annals of Statistics*, vol. 33, no. 4, pp. 1538–1579, 2005.

[17] S. Rosset, J. Zhu, and T. Hastie, "Boosting as a regularized path to a maximum margin classifier," *J. Mach. Learn. Res.*, vol. 5, pp. 941–973, Dec. 2004.

[18] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[19] J. H. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, no. 405, pp. 165–175, 1989.

[20] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *J. Mach. Learn. Res.*, vol. 9, pp. 485–516, Jun. 2008.

[21] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, Jul. 2008.

[22] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[23] J. Malick, "A dual approach to semidefinite least-squares problems," *SIAM J. Matrix Anal. Appl.*, vol. 26, no. 1, pp. 272–284, Jan 2005.

[24] H. Qi and D. Sun, "A quadratically convergent newton method for computing the nearest correlation matrix," *SIAM J. Matrix Anal. Appl.*, vol. 28, no. 2, pp. 360–385, Jun. 2006.