Density-Based Logistic Regression

Wenlin Chen
Department of Computer Science and Engineering
Washington University, St. Louis, USA
wenlinchen@wustl.edu

Yixin Chen
Department of Computer Science and Engineering
Washington University, St. Louis, USA
chen@cse.wustl.edu

Yi Mao, Baolong Guo
Department of Mechanical Engineering
Xidian University, Xi'an, China
{ymao,blguo}@xidian.edu.cn

ABSTRACT

This paper introduces a nonlinear logistic regression model for classification. The main idea is to map the data to a feature space based on kernel density estimation. A discriminative model is then learned to optimize the feature weights as well as the bandwidth of a Nadaraya-Watson kernel density estimator. We then propose a hierarchical optimization algorithm for learning the coefficients and kernel bandwidths in an integrated way. Compared to other nonlinear models such as kernel logistic regression (KLR) and SVM, our approach is far more efficient since it solves an optimization problem with a much smaller size. Two other major advantages are that it can cope with categorical attributes in a unified fashion and naturally handle multi-class problems. Moveover, our approach inherits from logistic regression good interpretability of the model, which is important for clinical applications but not offered by KLR and SVM. Extensive results on real datasets, including a clinical prediction application currently under deployment in a major hospital, show that our approach not only achieves superior classification accuracy, but also drastically reduces the computing time as compared to other leading methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-Data Mining; J.3 [**Computer Applications**]: Life and medical Sciences

Keywords

Nonlinear classification; logistic regression; density estimation; medical prediction

1. INTRODUCTION

Classification is a central and critical task in the area of data mining. Many classification models have been proposed, such as support vector machine (SVM), logistic regression (LR), kernel logistic regression (KLR), decision tree (DT), and naive Bayes (NB) classifier. There are a few key aspects regarding any classifier: 1) nonlinear separation ability, 2) support for mixed data types (numerical and categorical), 3) efficiency, 4) interpretability of model, and 5) support for multiway (as opposed to binary) classification. Few methods can competently achieve all of them.

An application that motivates this research is patient classification for early clinical warning, an NIH project currently under clinical trial at the Barnes Jewish Hospital, one of the largest hospitals in the United States. As we reported earlier [13, 14], we classify patients for categorical outcomes of certain disease or clinical event based on real-time data such as vital signs. This application entails mixed data types and requires good interpretability of model. Our previous study has chosen LR as the classifier for this application [13, 14].

Kernel-based SVM has good nonlinear separation ability. However, its output model is hard to interpret, which severely limits its use in some domains such as clinical warning. Moreover, the outcome of SVM is score-based rather than confidence-based. The score output from SVM for one task is not comparable to another task and often makes little sense to end users. Finally, SVM is inherently a binary classifier. Although there have been efforts to extend SVM to multiway classification, it is known that these methods have many limitations [1]. For example, the popular one-versus-the-rest method suffers from issues such as: the training is based on imbalanced datasets, and an instance may be assigned to multiple classes.

LR uses a linear weighted sum of the input attributes. LR has some advantages over SVM as it is often more efficient, and it is easier to extend LR to multiway classification. Moreover, LR has good interpretability: its confidence-based output is meaningful and comparable, and the weights associated with each feature can indicate the most important factors. However, LR is a linear classifier with a linear decision boundary. Extending LR by the kernel trick results in KLR [11, 20], which enables nonlinear classification. Unlike SVM, KLR cannot be formulated as a quadratic programming problem and iterative nonlinear optimization algorithms such as quasi-Newton methods are needed. As a result, the time complexity for training KLR is high. Moreover, KLR no longer offers good interpretability.

SVM, LR and KLR rely on numerical attributes and cannot naturally handle categorical attributes. When handling categorical attributes, they need to first transform those attributes to numerical features. A typical way is using m numbers to represent an m-category attribute [6]. Only one of the m numbers is 1 with the others being 0. However, this

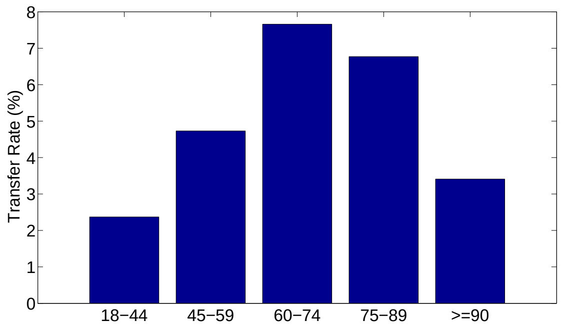

Figure 1: ICU transfer rate for various age groups.

kind of preprocessing dramatically increases the number of features.

In this paper, we propose a novel density-based logistic regression (DLR) model that well addresses all the five aspects to some extent. A key drawback of LR is that it assumes a *monotonic* relationship between every numerical attribute and the class probability, due to its linear formulation. However, such monotonicity is often false. For example, Figure 1 shows the ICU transfer rate versus the patients' age group. We can see that there is a non-monotonic relationship. Based on kernel density estimation (KDE), DLR can model non-monotonic and nonlinear relationships. Moreover, unlike LR, DLR can handle datasets with mixed data types since its nonlinear transformation can be applied to numerical and categorical attributes in a unified way. DLR also inherits the advantages of LR, including time efficiency, good interpretability, and support for multiway classification, which are not offered by other nonlinear methods such as SVM and KLR.

This paper contains the following contributions.

1) We propose DLR, a novel nonlinear classification model, by integrating Bayesian analysis and kernel density estimation into LR. DLR is much more efficient than other nonlinear models and can naturally handle mixed data types. It also offers good interpretability and support for multiway classification.

2) We develop a hierarchical optimization algorithm for training DLR, which automatically learns free parameters in the model under a maximum likelihood framework. This optimization formulation not only learns the coefficients in DLR, but also provides a way to automatically select the kernel bandwidth in the Nadaraya-Watson estimator, which is absent in previous work. This makes DLR a self-tuning model without any free parameters.

3) We extend the DLR model to multiway classification based on the same theory for binary LR. The hierarchical optimization algorithm for binary DLR can also be applied to the multiway case.

4) We conduct extensive evaluation on a large collection of biomedical datasets, mixed-type datasets, and a real patient dataset for clinical prediction. The results show that DLR compares favorably against other nonlinear methods including SVM and KLR in terms of classification quality. Moreover, DLR has much better efficiency and scalability – requiring drastically less, often orders of magnitude lower, training time than other kernel-based methods.

2. DENSITY-BASED LOGISTIC REGRESSION (DLR) MODEL

In this section, we first discuss the basics and limitations of LR and then describe our new DLR model.

2.1 Limitations of LR

Assume we are given a dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}$, $i = 1, \cdots, N$, $\mathbf{x}_i \in \mathcal{R}^D$, and $y_i \in \{0, 1\}$. Let the input vector be $\mathbf{x} = (x_1, \cdots, x_D)$, and the class label y be binary: y can be either 1 or 0. LR is based on the following probability model:

$$\begin{aligned} p(y=1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x}) =& \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})} \\ =& \frac{1}{1+\exp(-\sum_{d=0}^{D} w_d x_d)}, \end{aligned} \quad (1)$$

where $\mathbf{w}$ is a vector of weights that need to be learned. Note that $x_0 = 1$ represents a constant term. LR uses a maximum likelihood optimization to learn the weight vector $\mathbf{w}$.

Define

$$\tau(\mathbf{x}) = \ln \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})}. \quad (2)$$

Lemma 1. *LR models the following distribution of data:*

$$\tau(\mathbf{x}) = \sum_{d=0}^{D} w_d x_d \quad (3)$$

Proof. Based on the Bayesian rule, we find the probability

$$\begin{aligned} p(y=1|\mathbf{x}) &= \frac{p(y=1|\mathbf{x})}{p(y=1|\mathbf{x}) + p(y=0|\mathbf{x})} & (4) \\ &= \frac{1}{1+\exp(-\tau(\mathbf{x}))} & (5) \end{aligned}$$

The lemma follows by comparing (1) with (5). ■

From Lemma 1, we see that in a LR model: a) if $w_d > 0$, then $p(y = 1|\mathbf{x})$ and $\tau(\mathbf{x})$ increase as x_d increases, and b) if $w_d < 0$, then $p(y = 1|\mathbf{x})$ and $\tau(\mathbf{x})$ decrease as x_d increases.

Therefore, a severe drawback of LR is that it is reasonable only if there is a monotonic relationship between $p(y = 1|\mathbf{x})$ and x_d. However, this condition is often violated. For example, as we show in Figure 1, the probability of ICU transfer is not in a monotonic relationship with the *age* attribute. KLR and SVM can model non-monotonic relationships between $p(y = 1|\mathbf{x})$ and x_d. However, KLR and SVM have many other limitations such as high computational cost and low interpretability.

2.2 DLR and its feature transformation

We derive an attribute transformation for LR based on Bayesian rules. For simplicity of presentation, we first discuss the case where the class label y is binary: $y \in \{0, 1\}$ and extend it to the multiway case later. The proposed DLR model follows the parametric form of LR in (1), but transforms each attribute x_d to a feature $\phi_d(\mathbf{x})$:

$$\begin{aligned} p(y=1|\mathbf{x}) = \sigma(\mathbf{w}^T\phi) =& \frac{1}{1+\exp(-\mathbf{w}^T\phi)} \\ =& \frac{1}{1+\exp(-\sum_{d=0}^{D} w_d \phi_d(\mathbf{x}))}, \end{aligned} \quad (6)$$

where $\phi_0 = 1$. For each $d = 1, \cdots, D$, the proposed DLR feature transformation ϕ_d is

$$\phi_d(\mathbf{x}) = \ln \frac{p(y=1|x_d)}{p(y=0|x_d)} - \frac{D-1}{D} \ln \frac{p(y=1)}{p(y=0)}. \tag{7}$$

The first term in (7) relates to the likelihood of $y = 1$ given x_d. The second term $\frac{D-1}{D} \ln \frac{p(y=1)}{p(y=0)}$ can be viewed as a bias of the dataset. Hence, $\phi_d(\mathbf{x})$ gives an unbiased measurement of the quantity sought after by the LR formulation based on the Bayesian explanation.

Lemma 2. *If all the attributes of $\boldsymbol{x} = (x_1, \cdots, x_D)$ are conditionally independent given the label y, then:*

$$p(\mathbf{x}, y) = \prod_{d=1}^{D} \left(\frac{p(x_d, y)}{p(y)^{\frac{D-1}{D}}} \right) \tag{8}$$

Proof. Given that all variables are conditionally independent, we know $p(\mathbf{x}|y) = \prod_{d=1}^{D} p(x_d|y)$. It follows that

$$\begin{aligned} p(\mathbf{x}, y) &= p(\mathbf{x}|y)p(y) = p(y) \prod_{d=1}^{D} p(x_d|y) \\ &= \frac{\prod_{d=1}^{D} p(x_d|y)p(y)}{p(y)^{D-1}} = \frac{\prod_{d=1}^{D} p(x_d, y)}{p(y)^{D-1}} \\ &= \prod_{d=1}^{D} \left(\frac{p(x_d, y)}{p(y)^{\frac{D-1}{D}}} \right) \end{aligned} \tag{9}$$

■

Theorem 1. *For a dataset, if all the attributes of $\boldsymbol{x}$ are conditionally independent given the label y, then the DLR function (6) with $w_0 = 0$ and $w_d = 1$ for $d = 1, \cdots, D$ models the true distribution of data.*

Proof. Following a similar proof as in Lemma 1, we know that DLR models the following distribution of data:

$$\tau(\mathbf{x}) = \sum_{d=0}^{D} w_d \phi_d(\mathbf{x}) = \sum_{d=1}^{D} \phi_d(\mathbf{x}). \tag{10}$$

It remains to show that (10) is true for the given dataset. Based on conditional independence and Lemma 2, we have

$$\begin{aligned} \tau(\mathbf{x}) &= \ln \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})} = \ln \frac{p(\mathbf{x}, y=1)}{p(\mathbf{x}, y=0)} \\ &= \ln \frac{\prod_{d=1}^{D} \left(p(x_d, y=1)/p(y=1)^{\frac{D-1}{D}} \right)}{\prod_{d=1}^{D} \left(p(x_d, y=0)/p(y=0)^{\frac{D-1}{D}} \right)} \\ &= \ln \frac{\prod_{d=1}^{D} \left(p(y=1|x_d)p(x_d)/p(y=1)^{\frac{D-1}{D}} \right)}{\prod_{d=1}^{D} \left(p(y=0|x_d)p(x_d)/p(y=0)^{\frac{D-1}{D}} \right)} \\ &= \ln \frac{\prod_{d=1}^{D} \left(p(y=1|x_d)/p(y=1)^{\frac{D-1}{D}} \right)}{\prod_{d=1}^{D} \left(p(y=0|x_d)/p(y=0)^{\frac{D-1}{D}} \right)} \\ &= \sum_{d=1}^{D} \left(\ln \frac{p(y=1|x_d)}{p(y=0|x_d)} - \frac{D-1}{D} \ln \frac{p(y=1)}{p(y=0)} \right) \\ &= \sum_{d=1}^{D} \phi_d(\mathbf{x}) \end{aligned}$$

■

Theorem 1 shows that, under the conditional independence assumption, DLR with $w_0 = 0$ and $w_d = 1$ for $d = 1, \cdots, D$ perfectly models the distribution of data.

We also observe some connections between the naive Bayes (NB) model and DLR. NB first models $p(x_d|y)$ and then uses the Bayesian rule to figure out $p(y|\mathbf{x})$, while DLR directly models $p(y|x_d)$. Although DLR makes the same assumption of conditional independence as NB does, it is not rigidly tied to this assumption as NB does. NB can be viewed as a special case of DLR. DLR is more general as it incorporates the discriminative ability of LR and the generative power of NB.

2.3 Kernel density estimation for ϕ

We have proposed the transformation ϕ for DLR in (7). Now we estimate the probabilities in (7). Given training data $\mathcal{D} = \{\mathbf{x}_i, y_i\}$, $i = 1, \cdots, N$, we need to estimate $\phi_d(\mathbf{x})$ for each d. Divide $\mathcal{D}$ into two subsets $\mathcal{D}_1$ and $\mathcal{D}_0$, which contain data samples with $y = 1$ and $y = 0$, respectively.

Two quantities are needed for computing $\phi_d(\mathbf{x})$ in (7): $p(y|x_d)$ and $p(y)$. $p(y)$ can be estimated by simply counting the proportion of y in $\mathcal{D}$, i.e.

$$\hat{p}(y = k) = \frac{|\mathcal{D}_k|}{N}, \quad k = 0, 1. \tag{11}$$

To estimate $p(y|x_d)$, we distinguish the cases of categorical and numerical attributes.

If an attribute x_d takes categorical values, $p(y = k|x_d)$ can be estimated by the proportion of samples with $y = k$ among all the samples whose d^{th} attribute is x_d. Thus, it can be easily computed using simple counting:

$$\hat{p}(y = k|x_d) = \frac{|\mathcal{D}_k \bigcap D_{x_d}|}{|D_{x_d}|}, k = 0, 1, \tag{12}$$

where $D_{x_d} = \{\mathbf{x}_i \mid x_{i,d} = x_d, i = 1, \cdots, N\}$ is the set of samples in $\mathcal{D}$ whose d^{th} attribute is x_d. Note that we use $x_{i,d}$ to denote the d^{th} attribute of the i^{th} sample $\mathbf{x}_i$. Substituting (11) and (12) into (7), we have:

$$\phi_d(\mathbf{x}) = \ln \frac{|\mathcal{D}_1 \bigcap D_{x_d}|}{|\mathcal{D}_0 \bigcap D_{x_d}|} - \frac{D-1}{D} \ln \frac{|\mathcal{D}_1|}{|\mathcal{D}_0|}. \tag{13}$$

If an attribute x_d takes numerical values, we propose to use a Nadaraya-Watson type kernel density estimator to estimate $p(y = k|x_d), k = 0, 1$.

According to the Nadaraya-Watson estimator [1], we have:

$$\hat{p}(y = k|x_d) = \frac{\sum_{i \in \mathcal{D}_k} K(\frac{x_d - x_{i,d}}{h_d})}{\sum_{i=1}^{N} K(\frac{x_d - x_{i,d}}{h_d})} \tag{14}$$

where $K(x)$ is a kernel function satisfying $K(x) \geq 0$ and $\int K(x)dx = 1$, and $h_d > 0$ is a parameter called the *bandwidth* of the kernel density function. In this paper, we choose the Gaussian kernel for $K(x)$, namely,

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2}). \tag{15}$$

Substituting the estimates in (14) into (7), we have our transformation for the d^{th} dimension:

$$\phi_d(\mathbf{x}) = \ln \frac{\sum_{i \in \mathcal{D}_1} K(\frac{x_d - x_{i,d}}{h_d})}{\sum_{i \in \mathcal{D}_0} K(\frac{x_d - x_{i,d}}{h_d})} - \frac{D-1}{D} \ln \frac{|\mathcal{D}_1|}{|\mathcal{D}_0|}. \tag{16}$$

Using the Gaussian kernel, we have:

$$\phi_d(\mathbf{x}) = \ln \frac{\sum_{i\in\mathcal{D}_1} \exp(-\frac{(x_d - x_{i,d})^2}{2h_d^2})}{\sum_{i\in\mathcal{D}_0} \exp(-\frac{(x_d - x_{i,d})^2}{2h_d^2})} - \frac{D-1}{D} \ln \frac{|\mathcal{D}_1|}{|\mathcal{D}_0|}. \quad (17)$$

(17) gives the full closed form of $\phi_d(\mathbf{x})$ for a numerical x_d.

In essence, we use a Bayesian explanation of LR to derive a "reasonable" feature transformation in (7), and then use a kernel density estimator to compute the conditional probabilities in (7).

3. LEARNING ALGORITHM FOR DLR

Now we develop an algorithm for learning the parameters in the DLR model, including the weights $\mathbf{w}$ and the bandwidth h_d in the Nadaraya-Watson estimator for each numerical attribute x_d.

For each input $\mathbf{x}_i$, define $b_i = p(y_i = 1|\mathbf{x}_i)$ as given in (6). The objective of our optimization is to minimize the negative logarithm of the likelihood without any constraints, also known as the cross-entropy error function:

$$\begin{aligned} E(\mathbf{w}, \mathbf{h}) &= -\ln p(\mathbf{y}|\mathbf{w}, \mathbf{h}) \\ &= -\sum_{i=1}^{N} \{y_i \ln b_i + (1-y_i)\ln(1-b_i)\}, \end{aligned} \quad (18)$$

where $\mathbf{h}$ is the vector of all the $h_d, 1 \le d \le D$, where x_d is a numerical attribute (since categorical attributes do not need a Nadaraya-Watson estimator).

We minimize $E(\mathbf{w}, \mathbf{h})$ by performing gradient descent in the $\mathbf{w}$ and $\mathbf{h}$ spaces, based on the training set and validation set, respectively. We first compute the derivative of E with respect to h_d. The derivatives of E with respect to $\mathbf{w}$ are well studied in LR and easy to compute.

3.1 Kernel bandwidth selection

In each Nadaraya-Watson estimator for x_d, $d = 1, \cdots, D$, we need to set its bandwidth h_d. A common method in previous works use rules-of-thumb to set a heuristic h_d values. A popular one is the Silverman's rule of thumb [16]:

$$h_d^* = 1.06\sigma N^{-1/5}, \quad (19)$$

where σ is the standard deviation of x_d.

Although such rules-of-thumb often give solid performance, based on the DLR framework, we can in fact derive a novel way to automatically choose optimal h_d. Such automatic tuning is absent in previous work. We propose to find the h_d that minimizes E in (18). For this minimization, we first need to find $\frac{\partial E}{\partial h_d}$. Let $r_d = -\frac{1}{2h_d^2}$, according to (16), we have

$$\begin{aligned} \phi_d(\mathbf{x}) =& \ln \frac{\sum_{i\in\mathcal{D}_1} \exp(r_d(x_d - x_{i,d})^2)}{\sum_{i\in\mathcal{D}_0} \exp(r_d(x_d - x_{i,d})^2)} - \frac{D-1}{D} \ln \frac{|\mathcal{D}_1|}{|\mathcal{D}_0|} \\ =& g_1 - g_0 - S, \end{aligned} \quad (20)$$

where

$$g_j = \ln \sum_{i\in\mathcal{D}_j} \exp(r_d(x_d - x_{i,d})^2), \quad j = 0, 1, \quad (21)$$

$$\text{and} \quad S = \frac{D-1}{D} \ln \frac{|\mathcal{D}_1|}{|\mathcal{D}_0|}. \quad (22)$$

Thus, the derivative of $\phi_d(\mathbf{x})$ over r_d is

$$\frac{\partial \phi_d(\mathbf{x})}{\partial r_d} = \frac{\partial g_1}{\partial r_d} - \frac{\partial g_0}{\partial r_d}. \quad (23)$$

Moreover, for $j = 0, 1$,

$$\begin{aligned} \frac{\partial g_j}{\partial r_d} =& \frac{\sum_{i\in\mathcal{D}_j} [(x_d - x_{i,d})^2 \cdot \exp(r_d(x_d - x_{i,d})^2)]}{\sum_{i\in\mathcal{D}_j} \exp(r_d(x_d - x_{i,d})^2)} \\ =& \frac{\sum_{i\in\mathcal{D}_j} [(x_d - x_{i,d})^2 \cdot K(\frac{x_d - x_{i,d}}{h_d})]}{\sum_{i\in\mathcal{D}_j} K(\frac{x_d - x_{i,d}}{h_d})} \end{aligned} \quad (24)$$

Now we calculate the derivatives of $\ln b_i$ and $\ln(1 - b_i)$ over r_d which are needed in $\frac{\partial E}{\partial r_d}$. According to (1), we have

$$\begin{aligned} \frac{\partial \ln b_i}{\partial r_d} =& \frac{\exp(-\sum_{d=1}^{D} w_d \phi_d(\mathbf{x}_i))}{1 + \exp(-\sum_{d=1}^{D} w_d \phi_d(\mathbf{x}_i))} w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d} \\ =& (1 - b_i) w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d}, \end{aligned} \quad (25)$$

and

$$\begin{aligned} \ln(1 - b_i) =& \ln \frac{\exp(-\sum_{d=1}^{D} w_d \phi_d(\mathbf{x}_i))}{1 + \exp(-\sum_{d=1}^{D} w_d \phi_d(\mathbf{x}_i))} \\ =& -\sum_{d=1}^{D} w_d \phi_d(\mathbf{x}_i) + \ln b_i, \end{aligned} \quad (26)$$

$$\frac{\partial \ln(1 - b_i)}{\partial r_d} = -w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d} + \frac{\partial \ln b_i}{\partial r_d} = -b_i w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d} \quad (27)$$

Using (27) and (25), we get the following derivative of (18):

$$\begin{aligned} \frac{\partial E}{\partial r_d} =& -\sum_{i=1}^{N} \{y_i \frac{\partial \ln b_i}{\partial r_d} + (1 - y_i) \frac{\partial \ln(1 - b_i)}{\partial r_d}\} \\ =& \sum_{i=1}^{N} (b_i - y_i) w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d}. \end{aligned} \quad (28)$$

Then, since $r_d = -\frac{1}{2h_d^2}$, we get:

$$\frac{\partial E}{\partial h_d} = \frac{\partial E}{\partial r_d} \cdot \frac{\partial r_d}{\partial h_d} = \frac{1}{h_d^3} \sum_{i=1}^{N} (b_i - y_i) w_d \frac{\partial \phi_d(\mathbf{x}_i)}{\partial r_d}. \quad (29)$$

Finally, we substitute (23) and (24) into (29) and get the full closed-form expression of $\frac{\partial E}{\partial h_d}$.

3.2 Hierarchical optimization

We have obtained first-order derivative $\frac{\partial E}{\partial h_d}$ in (29). However, the second-order derivatives for $\mathbf{h}$ are hard to compute. And optimizing both $\mathbf{w}$ and $\mathbf{h}$ on the same set would simply result in a trivial solution where all $\mathbf{h}$ are 0. Therefore, it is difficult to perform Newton's method in the joint space of $\mathbf{w}$ and $\mathbf{h}$. To address this issue, we propose to learn $\mathbf{w}$ and $\mathbf{h}$ separately by a hierarchical optimization framework, shown in Algorithm 1, which contains two levels of optimization: an outer loop which optimizes $\mathbf{h}$ using simple gradient descent on validation set, and an inner loop which optimizes $\mathbf{w}$ using Newton's method on training set under fixed $\mathbf{h}$.

Similar to tuning free hyperparameters in SVM [2], we divide the dataset into the training set, validation set, and test

Algorithm 1 Hierarchical optimization for DLR learning

```
1: Initialize $\mathbf{h}$ using (19)
2: repeat                                        ▷ outer loop: optimize $\mathbf{h}$
3:     Assemble the feature matrix $\mathbf{\Phi}$ under $\mathbf{h}$
4:     repeat                  ▷ inner loop: fix $\mathbf{h}$ and optimize $\mathbf{w}$
5:         $\mathbf{w} \leftarrow \mathbf{w} - \frac{\nabla_{\mathbf{w}} E}{\nabla^2_{\mathbf{w}} E}$                    ▷ on the training set
6:     until $\mathbf{w}$ converges
7:     for $d = 1$ to $D$ do                  ▷ fix $\mathbf{w}$ and update $h_d$
8:         if $x_d$ is a numerical attribute then
9:             $h_d \leftarrow h_d - \gamma \frac{\partial E}{\partial h_d}$              ▷ on the validation set
10:        end if
11:    end for
12: until $\mathbf{h}$ converges
```

set. In the outer level, at each iteration, the feature matrix $\mathbf{\Phi}$, composed of $\phi(\mathbf{x}_i)$ for $i = 1, \cdots, D$, is updated based on the new $\mathbf{h}$ (Line 3). If a variable $\mathbf{x}_i$ is categorical, $\phi(\mathbf{x}_i)$ is computed by (13). For a numerical variable x_d, we use the kernel density estimation in (16) to compute its feature $\phi_d(\mathbf{x}_i)$. Then, entering the inner level, we fix $\mathbf{h}$ and optimize $\mathbf{w}$ by Newton's method using the training set (Lines 4-6). In fact, given $\mathbf{\Phi}$, we can use any standard LR package to implement Lines 4-6 and leverage its mature implementation. Finally, we optimize $\mathbf{h}$ for numerical attributes by performing gradient descent along the direction given in (29) using the validation set (Lines 7-11).

From Algorithm 1, we can derive two variants of DLR. We call the complete algorithm DLR-h since it automatically chooses $\mathbf{h}$. If we fix $\mathbf{h}$ at its initial rules-of-thumb values given in (19) and skip the optimization steps in Lines 7-11, we call this algorithm DLR. In the experimental result section, we will evaluate both DLR and DLR-h.

3.3 Extension to multiway classification

It is straightforward to extend DLR to multiway problems. We outline the main steps here. Assume there are C classes. We have, for $k = 1, \cdots, C$, the DLR model is as follows:

$$p(y = k|\mathbf{x}) = \frac{\exp(\mathbf{w}_k^T \phi_k(\mathbf{x}))}{\sum_{j=1}^{C} \exp(\mathbf{w}_j^T \phi_j(\mathbf{x}))} \tag{30}$$

where $\mathbf{w}_k = (w_{k,1}, \cdots, w_{k,D})$ is the weight coefficient vector for class k and $\phi_k = (\phi_{k,1}, \cdots, \phi_{k,D})$ is feature transformation for class k, defined as:

$$\phi_{k,d}(\mathbf{x}) = \ln p(y = k|x_d) - \frac{D-1}{D} \ln p(y = k). \tag{31}$$

If x_d is a numerical attribute, we estimate $p(y = k|x_d)$ by a Nadaraya-Watson estimator,

$$p(y = k|x_d) = \ln \frac{\sum_{i \in \mathcal{D}_k} \exp(-\frac{(x_d - x_{i,d})^2}{h_d^2})}{\sum_{i=1}^{N} \exp(-\frac{(x_d - x_{i,d})^2}{h_d^2})}, \tag{32}$$

where $\mathcal{D}_k \subseteq \mathcal{D}$ is the subset of data in class k.

Like binary DLR, multiway DLR learns $\mathbf{w}$ and $\mathbf{h}$ by minimizing the negative logarithm of likelihood:

$$E(\mathbf{w}, \mathbf{h}) = -\sum_{i=1}^{N} \sum_{k=1}^{C} 1_{y_i = k} \ln p(y_i = k|\mathbf{x}_i) \tag{33}$$

where $1_{y_i=k}$ is 1 when $y_i = k$ and 0 otherwise. Let $b_{i,k} = p(y_i = k|\mathbf{x}_i)$, we have:

$$\frac{\partial \ln b_{i,k}}{\partial r_d} = \sum_{j=1}^{C} \left(1_{(j=k)} - b_{i,j}\right) w_j^{(d)} \frac{\partial \phi_{j,d}(\mathbf{x}_i)}{\partial r_d} \tag{34}$$

As in (23) and (24), we have

$$\begin{aligned}\frac{\partial \phi_{j,d}(\mathbf{x}_i)}{\partial r_d} =& \frac{\sum_{i \in \mathcal{D}_j} [(x_d - x_{i,d})^2 \cdot \exp(r_d(x_d - x_{i,d})^2)]}{\sum_{i \in \mathcal{D}_j} \exp(r_d(x_d - x_{i,d})^2)} \\ &- \frac{\sum_{i=1}^{N} [(x_d - x_{i,d})^2 \cdot \exp(r_d(x_d - x_{i,d})^2)]}{\sum_{i=1}^{N} \exp(r_d(x_d - x_{i,d})^2)}\end{aligned}$$

Some simple calculation based on (33) and (34) gives:

$$\begin{aligned}\frac{\partial E}{\partial h_d} &= \frac{\partial E}{\partial r_d} \frac{\partial r_d}{\partial h_d} \\ &= \frac{1}{h_d^3} \sum_{i=1}^{N} \sum_{k=1}^{C} \sum_{j=1}^{C} y_{i,k} \left(b_{i,j} - 1_{(j=k)}\right) w_{j,d} \frac{\partial \phi_{j,d}(\mathbf{x}_i)}{\partial r_d}\end{aligned}$$

Given $\frac{\partial E}{\partial h_d}$, Algorithm 1 can also be used for training the multiway DLR model.

4. DISCUSSION: A NEW PROBABILISTIC DISCRIMINATIVE KERNEL

In this section, we discuss the DLR model from the perspective of kernel methods and illustrate that the DLR kernel is a valid and good kernel. We also point out its difference from some existing kernels.

It is known that the LR and SVM models can be studied under a unified view [1, 11]. They both pursue the goal of minimizing a regularized error function as follows:

$$\min_{\mathbf{w}} E = \frac{1}{2}||\mathbf{w}||^2 + C \sum_i g(-y_i \cdot \mathbf{w}\mathbf{x}_i) \tag{35}$$

The regularized LR model adopts the logistic loss where

$$g(\eta) = \ln(1 + e^{\eta}), \tag{36}$$

and the SVM model utilizes the hinge loss, i.e.

$$g(\eta) = \max(1 - \eta, 0) \tag{37}$$

Using the Representer Theorem, both models can be combined with the kernel trick to gain the ability of nonlinear separation.

From this perspective, the proposed DLR model can be viewed as taking the logistic loss as in LR. However, DLR uses the following kernel:

$$\begin{aligned}k_{DLR}(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})\phi(\mathbf{x}') &= \sum_{d=1}^{D} \phi_d(\mathbf{x})\phi_d(\mathbf{x}') \\ &= \sum_{d=1}^{D} k_d(\mathbf{x}, \mathbf{x}')\end{aligned} \tag{38}$$

where the transformation ϕ_d here takes the form in (7) and $k_d(\mathbf{x}, \mathbf{x}') = \phi_d(\mathbf{x})\phi_d(\mathbf{x}')$.

To some extent, a kernel defines the similarity between two data samples. For example, the Gaussian kernel interprets similarity based on distance. According to (7) and as discussed in section 2.2, each ϕ_d indicates the "likelihood" of y being labelled 1 given x_d. Thus, the kernel in the DLR model defines the similarity of two instances by their "likelihood" of having the same label. Specifically, $k_d(\mathbf{x}, \mathbf{x}')$ represents the similarity of the d^{th} attribute of these two samples, and the overall similarity is the sum of similarities at each dimension. We can observe this fact by distinguishing the following cases:

- If x_d and x'_d both indicate high likelihood of being labelled 1 (or 0), then both $\phi_d(\mathbf{x})$ and $\phi_d(\mathbf{x}')$ are positive (or negative) and $k_d(\mathbf{x}, \mathbf{x}')$ is large.

- If x_d and x'_d indicate high likelihood of different labels, then $\phi_d(\mathbf{x})$ and $\phi_d(\mathbf{x}')$ will have different signs and $k_d(\mathbf{x}, \mathbf{x}')$ will be negative.

Obviously, the DLR kernel in (38) is a valid kernel since it is the inner product in the feature space and thus the kernel matrix is always positive definite. For a perfect kernel, we need $k(\mathbf{x}, \mathbf{x}')$ to satisfy the condition that $k(\mathbf{x}, \mathbf{x}') > 0$ when $y = y'$ and $k(\mathbf{x}, \mathbf{x}') < 0$ when $y \neq y'$ as stated in [4]. When a kernel strictly satisfies this condition, all the instances in the dataset would be perfectly classified without any error. From the above analysis, we see that the DLR kernel closely correlates to this condition, as it can be seen as a probabilistic implementation of this condition. Thus, the DLR kernel is not only a valid kernel, but also a good kernel.

Different from most existing kernels like the polynomial kernel and the Gaussian kernel, our kernel considers the label information by discriminative conditional probability, which leads to a better measurement of similarity, while existing kernels such as the Gaussian kernel do not consider the label information in y.

We can observe from (38) that the DLR kernel can be expressed in an additive form of kernels operating on each dimension, known as the additive kernel [12]. Thus, any classifiers using the DLR kernel inherit the good properties of additive kernels such as fast training and evaluation.

5. RELATED WORK

LR can also be studied from a Bayesian perspective, although exact Bayesian inference is infeasible and some approximation such as Laplace approximation is needed [9]. A dual algorithm for KLR using ideas similar to the SMO algorithm for SVM is proposed in [11]. The import vector machine (IVM) incorporates the loss function of KLR in a SVM framework [20]. IVM shares some advantages with LR, including the abilities to estimate the underlying probability and generalize to multi-class problems. Compared to these complex methods such as Bayesian logistic regression and IVM, our approach requires only a simple preprocessing to transform the input variables and requires solving a single unconstrained optimization problem as LR does. It also retains the advantages of LR over SVM.

Raina et al. proposed a hybrid generative/discriminative model for classification [15]. They also arrived at a form that transforms each variables x_d into a feature $\phi_d = \ln(p(x_d, y = 1) - \ln(p(x_d, y = 0)$. However, it only considers the case where x_d takes discrete values. Moreover, the probabilities $p(x_d, y = 1)$ and $p(x_d, y = 0)$ are learned using a NB model, which requires many samples and does not work for continuous x_d. In contrast, we use a smooth kernel density estimation which accommodates continuous x_d and requires fewer samples. We also optimize the bandwidth h_d automatically.

There are other related probabilistic kernels most of which aim at combining generative models with discriminative ones. Jaakkola and Haussler proposed using generative techniques in discriminative classification models [8]. They proposed to use the Fisher score from generative models to generate a kernel, which is in turn used in discriminative models. However, such generative kernels are only suitable for some domain-specific tasks and different tasks require different generative models. For example, the Fisher kernel of sequential data such as DNA is often derived from a hidden Markov model (HMM) [7], while the the kernel for text segmentation is often derived from LDA [17]. A more general probabilistic kernel is the marginalized kernel [10] for classifying graphs. However, it still requires a domain-specific generative model. A discriminative kernel is introduced in [19] where the mapping function is also based on $\ln(p(y|\mathbf{x}, \theta))$. However, it still relies on a particular generative model such as HMM to which θ belongs. Compared to these model-driven kernels [4], our DLR kernel does not assume any underlying models for data samples and uses the Nadaraya-Watson kernel density estimator to approximate $p(y|\mathbf{x})$, which results in a general classification approach that can be potentially used in a wide range of applications.

Our DLR model can also be considered a special case of the generalized additive model (GAM) [5]. GAM specifies a distribution $g(E(y)) = \sum_{d=1}^{D} f_d(x_d)$, where g is known as the link function and f_d is the basis function. The proposed DLR model is equivalent to using $\ln \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})}$ in (2) as the link function and $w_d\phi(x_d)$ as the basis function. In particular, DLR is closely related to additive logistic regression (ALR) [5], a GAM model for nonlinear classification. They use the same link function. However, rather than using splines as basis functions in ALR, we adopt the logarithm of kernel smooth functions as in (7). In addition, ALR is prohibitively time-consuming for high dimensional datasets due to the slow convergence of its backfitting algorithm. Also, It cannot naturally handle mixed data types as DLR does.

6. EXPERIMENTAL RESULTS

We conduct extensive experiments to evaluate DLR. We evaluate two versions of DLR: DLR with a fixed $\mathbf{h}$ given in (19) (denoted as DLR), and DLR with automatic tuning of $\mathbf{h}$ (DLR-h). For comparison, we also consider seven other classification methods, including logistic regression (LR), SVM with polynomial kernel (SVM-p) and RBF kernal (SVM-r), least square SVM [18] with polynomial kernel (LSSVM-p) and RBF kernal (LSSVM-r), and kernel logistic regressions with polynomial kernel (KLR-p) and RBF kernal (KLR-r). For all the datasets, we perform a standard normalization of the features. The normalization is especially helpful for SVMs. All algorithms are implemented in Matlab. The optimization in LR and DLR used the *minFunc* function in Matlab. The SVM package is implemented by Bottou and Lin in the Matlab Bioinformatics Toolbox, and the LSSVM package is the LS-SVMlab Toolbox. The regularization and kernel width parameters of SVM are also tuned on the validation set. All experiments are performed on a desktop with 2.67GHz CPU and 2G memory running Windows 7.

6.1 Illustrations on toy cases

For sanity check and illustration, we test on a simple 2-D dataset as shown Figure 2. It contains 1200 points, cho-

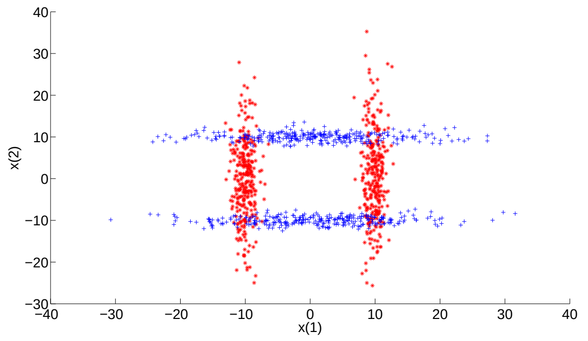

Figure 2: A simple 2-D dataset. The two classes are colored differently.

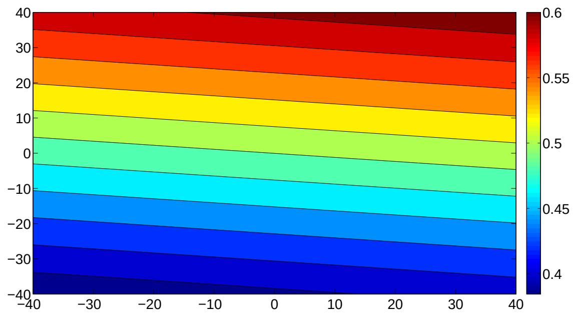

Figure 3: Probability output of LR. The decision boundary is at $p = 0.5$.

sen from 4 multivariate normal distributions with a mean of $[10, 0]$, $[-10, 0]$, $[0, 10]$ and $[0, -10]$, respectively. The covariances of former two distributions are $\sigma = [1, 0; 0, 100]$ while the last two distributions are $\sigma = [100, 0; 0, 1]$. The two classes are marked in different colors.

We plot the decision boundary of LR and DLR in Figure 3 and Fig. 4, respectively. We can see that the decision boundary of LR is linear, leading to a low accuracy of 51.3%. On the other hand, DLR gives a very nice nonlinear decision boundary and has an accuracy of 89.3%.

We also illustrate the process of Algorithm 1 which automatic tunes the kernel bandwidth h. As we explained before, a benefit of DLR compared to SVM is that it can naturally handle multi-class classification. In Figure 5, we illustrate the change of decision boundaries by tuning h on a three-class dataset. We can see that Algorithm 1 quickly converges to a near optimal h in only three iterations.

6.2 Results on numerical data

We test all the methods on five biomedical datasets from the UCI repository [3]. They have binary classes and numerical features. For each dataset, we run 100 experiments with different random 70/30 split for training/testing and report the averaged results. Table 1 and 2 compare the accuracy and AUC of various methods, respectively. We observe that DLR and DLR-h perform better than other methods in most datasets. We also see that, while DLR gives consistently good performance using the rule-of-thumb value of **h**, DLR-h achieves better accuracy than DLR on all the datasets.

Table 3 shows the training times. LR has the least running time while DLR is the second fastest and drastically more

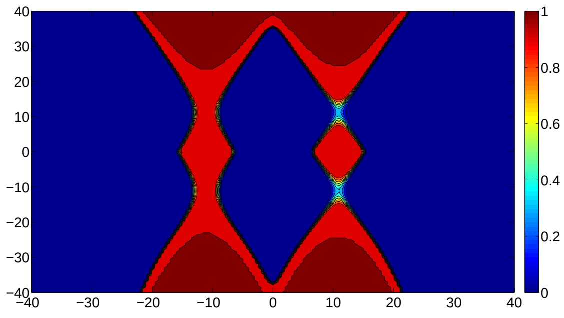

Figure 4: Probability output of DLR. The decision boundary is at $p = 0.5$.

efficient than KLR and SVM. DLR-h is slower than the DLR model since it tunes h, but it is still much faster than KLR and SVM in most cases. The main reason for the efficiency of DLR is that it optimizes a problem with D unknowns while kernel-based SVM and KLR search in a space with N unknowns, and typically $D \ll N$.

6.3 Results on categorical and mixed data

Another advantage of DLR is its ability to naturally handle categorical variables. We evaluate our algorithms on three datasets with categorical features from the UCI repository. For DLR and DLR-h, a categorical variable x is transformed into a numerical feature $\phi(x)$ based on density estimation in (16). For other methods, we use a typical way to handle categorical variables. For each variable that has m categories, we transform it into m numbers with only one of the numbers being 1 and the others being 0. Tables 4, 5, and 6 show the accuracy, AUC, and training time of various methods on these datasets, respectively. DLR and DLR-h have the same performance on *tic-tac* and *monk* datasets since both have categorical features only and there is no h_d to tune. SPECFT heart data have mixed types. We can see that DLR and DLR-h are high competitive with, if not better than, all other methods. However, a huge advantage of DLR and DLR-h is their time efficiency – they are orders of magnitude faster than other kernel based classifiers.

6.4 Real-world clinical prediction

Our research is motivated by a project in collaboration with the Barnes-Jewish Hospital (BJH), one of the largest hospitals in the US. Our task is to predict potential ICU transfers for hospitalized patients, based on 34 vital signs. We use a number of techniques to process the features, and the details of this process are reported in [13,14]. In a clinical trial at BJH, three algorithms, LR, SVM-r and DLR-h, are tested on more than 18,458 patients admitted from 01/14/11 to 04/23/12. Table 7 shows the results in terms of a few metrics that are important for clinical practice. We can see that DLR-h improves LR and SVM-r on most metrics. In particular, it significantly improves AUC, which measures the overall performance. Specificity and sensitivity are tradeoff that can be tuned in DLR-h by changing the threshold. We set DLR-h to have a specificity that is very close to SVM-r's. To study the scalability, we test different randomly chosen training datasets with various sizes. Figure 6 compares the training time with respect to the size of training data. We can see that DLR-h is much more efficient than SVM-r.

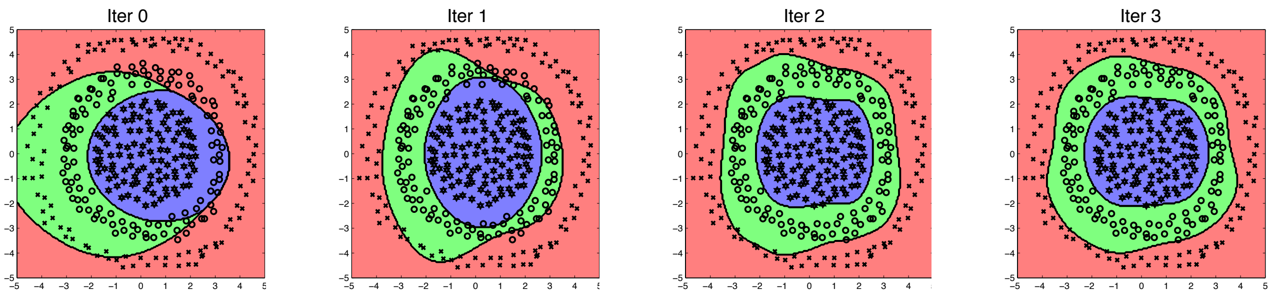

Figure 5: The process of optimizing the kernel density bandwidth h in DLR-h using Algorithm 1.

Table 1: Accuracy (%) of various methods on numerical data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
Wisconsin breast cancer	93.9	93.0	96.5	93.9	**97.3**	94.3	95.2	96.5	96.5
hepatitis	85.2	80.4	84.3	84.6	85.0	76.5	84.3	86.2	**88.2**
ionosphere	85.1	82.8	93.1	84.6	**94.4**	82.9	86.9	93.1	93.1
Cleveland heart disease	84.5	79.7	83.8	75.0	84.1	62.6	67.7	**85.1**	**85.1**
Pima Indians diabetes	74.7	74.0	75.0	70.8	77.7	72.0	74.3	75.5	**77.8**

Table 2: AUC of various methods on numerical data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
Wisconsin breast cancer	0.9923	0.9534	0.9835	0.9890	**0.9970**	0.6610	0.9980	0.9960	0.9960
hepatitis	0.8004	0.5000	0.5207	0.8049	0.8520	0.8732	0.8450	0.8580	**0.8781**
ionosphere	0.8711	0.9264	0.9635	0.7994	0.9650	0.9695	0.6220	**0.9890**	0.9695
Cleveland heart disease	0.8060	0.5000	0.6053	0.8217	0.8270	0.7810	**0.8790**	0.8128	0.8134
Pima Indians diabetes	0.8602	0.5000	0.6322	0.5949	0.8310	0.7665	0.8230	**0.8410**	0.7873

Table 3: Training time (ms) of various methods on numerical data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
Wisconsin breast cancer	46.8	1700.4	312.2	2620.8	2143.4	421.2	374.4	296.4	702.0
hepatitis	62.4	842.4	124.8	93.6	320.0	249.6	265.2	78.5	249.5
ionosphere	64.7	1154.4	1201.2	421.2	3842.6	358.8	234.0	124.8	546.0
Cleveland heart disease	31.2	1060.8	1154.8	499.2	830.0	390.0	280.8	123.0	604.2
Pima Indians diabetes	62.4	3042.0	3556.8	22245.7	6750.0	436.8	265.2	452.4	764.4

Table 4: Accuracy (%) of various methods on categorical and mixed data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
tic-tac	95.9	98.1	97.5	98.1	**98.4**	97.2	97.2	98.1	98.1
monk problem	69.8	97.3	97.3	97.3	**98.6**	97.1	96.5	97.3	97.3
SPECFT heart data	57.8	65.6	74.5	63.3	82.4	62.7	62.7	77.8	**87.6**

Table 5: AUC of various methods on categorical and mixed data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
tic-tac	0.9020	0.9489	0.9489	0.9462	**0.9516**	0.9510	0.9441	0.9457	0.9457
monk problem	0.9968	0.9912	0.9981	0.8747	0.9200	**0.9993**	0.9992	0.9979	0.9979
SPECFT heart data	0.7872	0.6839	0.5618	0.7691	0.8272	0.7277	0.7758	0.7898	**0.8488**

Table 6: Training time (ms) of various methods on categorical and mixed data.

Data set	LR	KLR-p	KLR-r	SVM-p	SVM-r	LSSVM-p	LSSVM-r	DLR	DLR-h
tic-tac	218.4	8907.6	1872.0	11548.7	3166.8	514.8	327.6	46.8	46.8
monk problem	187.2	1388.4	795.6	12230.4	567.2	452.4	374.4	31.3	31.2
SPECFT heart data	145.8	530.4	171.6	1716.0	2210.4	468.0	226.7	45.7	62.4

Table 7: Prediction performance on clinical data.

	AUC	Specificity	Sensitivity	PPV	NPV	Accuracy
LR	0.7714	0.9493	0.2963	0.2500	0.9594	0.9141
SVM-r	0.6715	**0.9520**	0.3909	0.2908	**0.9689**	0.9194
DLR-h	**0.8724**	0.9493	**0.4074**	**0.3143**	0.9654	**0.9201**

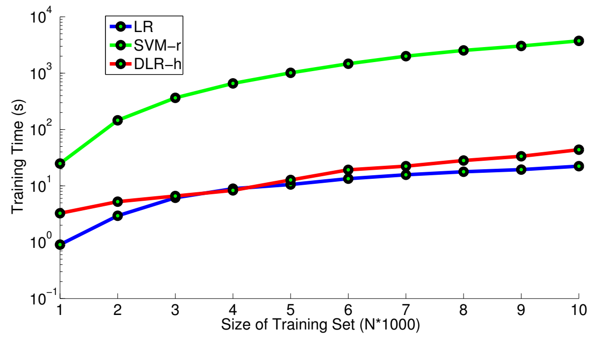

Figure 6: Training time on the clinical data. Note that the y-axis is in log scale.

7. CONCLUSIONS

In this paper, we have proposed a novel DLR model for classification. DLR integrates kernel density estimation and the discriminative power of logistic regression. DLR uses a novel nonlinear feature transformation derived from a Bayesian explanation of its parametric form, and a Nadaraya-Watson kernel density estimator for assessing the conditional probabilities in the transformation. We have also derived a hierarchical optimization algorithm for learning the model coefficients and kernel bandwidths in an integrated way. DLR competently supports nonlinear separation, efficient training, mixed data types, multiway classification, and good interpretability, a combination of advantages that is rarely found in existing methods. Extensive results on real-world numerical and categorical data show that, compared to other leading methods, DLR gives comparable and often better classification quality while being orders of magnitude more efficient.

We believe that, because of its unmatched performance, versatility, efficiency, and interpretability, DLR will become a popular general-purpose classification approach for many real applications.

Acknowledgment

This work is supported by the CNS-1017701 and CCF-1215302 grants from the National Science Foundation of USA.

8. REFERENCES

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] O. Chapelle, A. Polytechnique, and N. Cristianini. Choosing multiple parameters for support vector machines. In *Machine Learning*, 2002.

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

[4] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1):49–58, July 2003.

[5] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[6] C. W. Hsu, C. C. Chang, and C. J. Lin. *A practical guide to support vector classification*. Dept. CSIE, National Taiwan University, 2003.

[7] T. Jaakkola, M. Diekhans, and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. In *Proc. Int'l Conference on Intelligent Systems for Molecular Biology*, pages 149–158. AAAI Press, 1999.

[8] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Proc. NIPS*, pages 487–493, 1998.

[9] T. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *In Proceedings of the 1999 Conference on AI and Statistics*, 1999.

[10] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 321–328. AAAI Press, 2003.

[11] S. S. Keerthi, K. Duan, S. K. Shevade, and A. N. Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61(1-3):151–165, 2005.

[12] S. Maji and A. C. Berg. Max-margin additive classifiers for detection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 40–47. IEEE, 2009.

[13] Y. Mao, W. Chen, Y. Chen, C. Lu, M. Kollef, and T. Bailey. An integrated data mining approach to real-time clinical monitoring and deterioration warning. In *Proc. ACM SIGKDD*, 2012.

[14] Y. Mao, Y. Chen, G. Hackmann, M. Chen, C. Lu, M. Kollef, and T. Bailey. Early deterioration warning for hospitalized patients by mining clinical data. *International Journal of Knowledge Discovery in Bioinformatics*, 2(3):1–20, 2012.

[15] R. Raina, Y. Shen, A. Ng, and A. McCallum. Classification with hybrid generative/discriminative models. In *Proc. NIPS*, 2003.

[16] B. W. Silverman and P. J. Green. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[17] Q. Sun, R. Li, D. Luo, and X. Wu. Text segmentation with LDA-based fisher kernel. In *Proc. Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 269–272, 2008.

[18] J. A. K. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, and J. Vandewalle. *Least Squares Support Vector Machine*. World Scientific, Singapore, 2002.

[19] K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.-R. Müller. A new discriminative kernel from probabilistic models. *Neural Computing*, 14(10):2397–2414, 2002.

[20] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, pages 1081–1088, 2001.