A Simple Gaussian Kernel Classifier with Automated Hyperparameter Tuning

Kosuke Fukumori and Toshihisa Tanaka
Department of Electrical and Electronic Engineering,
Tokyo University of Agriculture and Technology, Tokyo, Japan
E-mail: fukumori17@sip.tuat.ac.jp, tanakat@cc.tuat.ac.jp Tel/Fax: +81-42-388-7123

***Abstract*—This paper establishes a fitting method for a kernel logistic regression model that uses generalized Gaussian kernel and its parameter optimization method. Kernel logistic regression is a classification model that uses kernel methods effectively. This is one of the methods to construct an effective non-linear system with a reproducing kernel Hilbert space (RKHS) induced from positive semi-definite kernels. Most classifiers that are combined with Gaussian kernel functions generally assume uncorrelatedness within the feature vectors. Thus, the Gaussian kernel consists of only two parameters (namely, mean and precision). In this paper, we propose a model using a generalized Gaussian kernel represented flexibly in each dimension of feature vector. In addition, the parameters of kernel are fully data-driven. For the fitting of proposed model, an ℓ_1-regularization is introduced to supress the number of support vectors. A numerical experiment showed that the classification performance of the proposed model is almost the same as RBF-SVM even though the proposed model has a small number of support vectors.**

I. Introduction

Deep learning (DL) shifted the paradigm of machine learning and artificial intelligence. A large number of layers with neurons can efficiently fit the data to extract features [1]. Thus, it is crucial to collect a large amount of data ("big data") to train millions of neurons to design an efficient network [2], [3]. In particular, for supervised learning, the big data should be always together with the label corresponding to each sample. Besides, high performance computing is a key technology to train such a large model of network. In practice, such an ideal situation is very limited. Many types of real-world data, such as medical data, environmental data, etc., cannot be big data [4], [5]. Sometimes they do not have enough labels, and sometimes they are not well formatted. Thus, it is still important to develop efficient methods of supervised learning for non-big data ("small data").

A well-established traditional approach to small data is support vector machine (SVM) [6]. The SVM cannot perform feature extraction by itself, but it is known that it works very efficiently with small data. A potential advantage of the kernel method is that the linear method can be applied directly to the nonlinear mapping of the feature vectors. Therefore, although the inner product on the high-dimension space to which this mapping belongs can not be calculated explicitly, it can be calculated using a kernel function by converting a high-dimension space to a reproducing kernel Hilbert space (RKHS). Thanks to the kernel method, SVM can achieve high estimation performance by expressing ability of nonlinear identification boundary and generalization ability based on geometry margin maximization [6], [7], [8].

Selection of kernel parameters is one of the important issues for generalization ability [9], [10]. The Gaussian kernel is widely used as a powerful kernel function for kernel classifiers [11]. Typically, the Gaussian kernel has two parameters: mean (vector) and precision (scalar). This means that the model assumes that the feature vectors are uncorrelatd. This is a strong assumption, because the observed samples are often correlated.

In this paper, we propose to use the generalized Gaussian function with hyperparameters of the mean vector and precision matrix as a kernel function. Adoping kernel logistic regression (KLR) with ℓ_1-regularizatoin, we establish fully data-driven estimation methods for the hyperparameters. The underlying idea behind the use of KLR is its differentiability of the cost function. This enables us to develop a Riemannian geometry-based gradient method to estimate the hyperparameters. Moreover, because of the ℓ_1-regularization, the model can prevent overfitting and supress the number of samples in the memory. Numerical experiments support the efficacy of the proposed method. For the experiment, we use 16 datasets of binary classification published on UCI Machine Learning Repository [12]. We verify the effectiveness of the proposed kernel optimization method by comparing the classification performance of SVM and the proposed method.

II. Kernel Logistic Regression and Support Vectors

Kernel Logistic Regression is a powerful and flexible nonlinear classification model which has discrimination performance equivalent to ohter traditional classifiers [13], [14]. KLR has advantages that it provides the posterior probability of the class and it is easy to differentiate the cost function. Here, we introduce KLR and a construction method of its support vectors using ℓ_1-regularization to propose our method. The symbols of definition which used in this paper are shown in Table I.

A. Kernel logistic regression in RKHS

KLR is an extended model of logistic regression (LR) that can solve nonlinear classification problem by the kernel

method [15], [16]. LR and KLR provide the posterior probability by the sigmoid function:

$$\phi(f(\boldsymbol{x}))=\frac{1}{1+\exp(-f(\boldsymbol{x}))} \tag{1}$$

as an activation function, where $f(\boldsymbol{x})$ is an inner product of the feature vector $\boldsymbol{x}$ and the weight vector $\boldsymbol{w}$ when build a model as LR, given as:

$$f(\boldsymbol{x})=\boldsymbol{w}^\top \boldsymbol{x}. \tag{2}$$

On the other hand, when build a model as KLR, let the $f(\boldsymbol{x})$ be the elements in RKHS. By introducing the representer theorem [15], $f(\boldsymbol{x})$ is given as:

$$f(\boldsymbol{x})=\sum_{i=1}^{N} h^{(i)} \mathcal{K}\left(\boldsymbol{x},\boldsymbol{x}^{(i)}\right). \tag{3}$$

Therefore, the probability $P(y=0|\boldsymbol{x})$ and $P(y=1|\boldsymbol{x})$ that the feature vector $\boldsymbol{x}$ is classified to the class label $y=0/1$ are obtained as:

$$P(y=0|\boldsymbol{x})=\frac{1}{1+\exp(-f(\boldsymbol{x}))}, \tag{4}$$

$$\begin{aligned} P(y=1|\boldsymbol{x}) &= 1-P(y=0|\boldsymbol{x}) \\ &= \frac{\exp(-f(\boldsymbol{x}))}{1+\exp(-f(\boldsymbol{x}))}. \end{aligned} \tag{5}$$

In this paper, to apply KLR as a classifier, we define the following classification rule:

$$\hat{y}=\underset{l\in\{0,1\}}{\operatorname{argmax}} P(y=l|\boldsymbol{x}). \tag{6}$$

By Eq. (6), KLR can predict the two values in the binary classification.

For the fitting of KLR, cross-entropy which is the log-likelihood of the Bernoulli distribution is employed as cost function:

$$\begin{aligned} J=-\frac{1}{N}\sum_{i=1}^{N}\Big[& y^{(i)}\log(\phi(f(\boldsymbol{x}))) \\ & +(1-y^{(i)})\log(1-\phi(f(\boldsymbol{x})))\Big]. \end{aligned} \tag{7}$$

Then, update the parameters to minimize this function. This is a minimization problem of the cost function with the kernel coefficient h as the explanatory variable [17].

B. Construction method of support vectors using ℓ_1-regularization

As shown in Eq. (3), the $f(\boldsymbol{x})$ is the sum of the kernel functions determined by the feature vectors of all the training data. This means that the higher the number N of training set is, the higher the computational cost is. One method to solve this problem is to construct a sparse model by removing unimportant kernel functions. In this section, the kernel functions that construct the model are called support vectors, and we describe a method to build a sparse model using ℓ_1-regularization.

TABLE I
DEFINITIONS OF SYMBOLS

N	Number of training set
i	Index for element of training set
j	Index for element of support vectors that constitute a model
k	Number of current learning iteration
$\boldsymbol{x}\in\mathbb{R}^m$	m-dimensional feature vector
$y\in\{0,1\}$	True value of the class label to which the feature vector $\boldsymbol{x}$ belongs
$\hat{y}\in\{0,1\}$	Predicted value of the class label to which the feature vector $\boldsymbol{x}$ belongs
$f(\cdot)$	Sum of an weighted feature vector or weighted kernel outputs
h	Kernel coefficient
$\phi(\cdot)$	Activation function
$\mathcal{K}(\cdot,\boldsymbol{x})$	Kernel function

The set of support vectors is a set of kernel functions represented by $\left\{\mathcal{K}(\cdot,\boldsymbol{x}^{(j)})\right\}_{j\in\mathcal{J}}$, where $\mathcal{J}:=\{j_1,j_2,...,j_r\}\subset\{0,1,...,N-1\}$ is the index of the support vectors. Here, we adopt an ℓ_1-regularization term into the cost function for updating of the kernel coefficient h. By the ℓ_1-regularization, the cost function J_{ℓ_1} is given as:

$$J_{\ell_1}=J+\lambda\sum_{j\in\mathcal{J}}\left|h^{(j)}\right|, \tag{8}$$

where λ is the regularization parameter. Since Eq. (8) is a non-convex function, the foward-backward splitting method (FOBOS) [18] is employed to minimizing this function. Thus, the update rule is given as:

$$h_{k+1}^{(j)}=\operatorname{sign}\left(\alpha_k^{(j)}\right)\left[\left|\alpha_k^{(j)}\right|-\lambda\eta_h\right]_+, \tag{9}$$

where

$$\alpha_k^{(j)}=h_k^{(j)}-\eta_h\left.\frac{\partial J(h)}{\partial h}\right|_{h=h_k^{(j)}},$$

η_h is a step size for the kernel coefficient h and k is a number of the current fitting step. After the updating of h, $\mathcal{K}(\cdot,\boldsymbol{x}^{(j)})$ is removed from the set of support vectors if $h_{k+1}^{(j)}=0$.

III. KLR FITTING USIGN GENERALIZED GAUSSIAN KERNEL

Most classifiers that are combined with Gaussian kernel functions generally assume uncorrelatedness within the feature vectors. Thus, the Gaussian kernel consists of only two parameters (namely, mean and precision). In this section, we propose a kernel logistic regression (KLR) model using a generalized Gaussian kernel function expressed as:

$$\mathcal{K}(\cdot,\,\boldsymbol{c};\,\boldsymbol{\Lambda})=\exp\left(-(\cdot-\boldsymbol{c})^\top\boldsymbol{\Lambda}(\cdot-\boldsymbol{c})\right), \tag{10}$$

where $\boldsymbol{\Lambda}\in\mathbb{R}^{m\times m}$ is an inverse covariance matrix, which is called the precision matrix, and $\boldsymbol{c}$ is called the mean of the Gaussian kernel function. Then, we establish a method to optimize both the precisions and the means at fitting the proposed KLR model. When the update of the kernel coefficients, the proposed method simultaneously update the

precisions and the means to increase the generalization ability. In addition, an ℓ_1-regularization is employed to reduce the support vectors.

In the proposed fitting method, the kernel parameters are defined as variables and updated every the k-th fitting step. Therefore, the sum of kernel functions $f(\boldsymbol{x})$ is given as:

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{j\in\mathcal{J}} h^{(j)} \mathcal{K}\Big(\boldsymbol{x},\, \boldsymbol{x}^{(j)};\, \boldsymbol{\Lambda}_k^{(j)}\Big) \\ &= \sum_{j\in\mathcal{J}} h^{(j)} \exp\big(-(\boldsymbol{x}-\boldsymbol{c}_k^{(j)})^\top \boldsymbol{\Lambda}_k^{(j)} (\boldsymbol{x}-\boldsymbol{c}_k^{(j)})\big), \end{aligned} \tag{11}$$

where $\boldsymbol{\Lambda}_k^{(j)}$ and $\boldsymbol{c}_k^{(j)}$ are the kernel precision and mean at k-th step, respectively.

A. Updating the kernel precision matrices

The kernel precision $\boldsymbol{\Lambda}$ must be a symmetric positive definite (SPD) matrix, which is denoted by $\mathbb{R}^{m\times m}$. In order to update these matrices while preserveing the SPD condition, we consider to employ the matrix exponentiated gradient (MEG) update [19] given as:

$$\begin{aligned} &\boldsymbol{\Lambda}_{k+1}^{(j)} \\ &= \exp\left(\log \boldsymbol{\Lambda}_k^{(j)} - \eta_{\boldsymbol{\Lambda}} \mathrm{sym}\left(\frac{\partial J(\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}}\right)\right), \end{aligned} \tag{12}$$

where $\eta_{\boldsymbol{\Lambda}}$ a step size for $\boldsymbol{\Lambda}$ and $\mathrm{sym}(\boldsymbol{X}) := (\boldsymbol{X}+\boldsymbol{X}^\top)/2$ denotes the symmetrization for $\boldsymbol{X}$. However, the computation of $\log \boldsymbol{\Lambda}_k^{(j)}$ might be unstable when the eigenvalues of $\boldsymbol{\Lambda}_k^{(j)}$ is very small. To avoid this possibility of becoming unstable, the proposed method normalizes $\boldsymbol{\Lambda}$ by the following function:

$$L_k^{(j)}(\boldsymbol{\Lambda}) = {\boldsymbol{\Lambda}_k^{(j)}}^{-1/2} \boldsymbol{\Lambda} {\boldsymbol{\Lambda}_k^{(j)}}^{-1/2}, \tag{13}$$

where $\boldsymbol{\Lambda}_k$ is a current matrix at the k-th step and is unchanged at this update time. For notational simplicity, we define the normalized $\boldsymbol{\Lambda}$ by Eq. (13) as $\boldsymbol{\Lambda}' := L_k^{(j)}(\boldsymbol{\Lambda}) = {\boldsymbol{\Lambda}_k^{(j)}}^{-1/2} \boldsymbol{\Lambda} {\boldsymbol{\Lambda}_k^{(j)}}^{-1/2}$. This implies $\boldsymbol{\Lambda} = {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} L_k^{(j)}(\boldsymbol{\Lambda}) {\boldsymbol{\Lambda}_k^{(j)}}^{1/2}$.

After the normalization by Eq. (13), the update rule for $\boldsymbol{\Lambda}'$ based on Eq. (12) can be written as:

$$\begin{aligned} &\boldsymbol{\Lambda}_{k+1}^{\prime(j)} \\ &= \exp\left(\log \boldsymbol{\Lambda}_k^{\prime(j)} - \eta_{\boldsymbol{\Lambda}'} \mathrm{sym}\left(\frac{\partial J(\boldsymbol{\Lambda}(\boldsymbol{\Lambda}'))}{\partial \boldsymbol{\Lambda}'}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}}\right)\right) \\ &= \exp\Bigg(\underbrace{\log\Big({\boldsymbol{\Lambda}_k^{(j)}}^{-1/2} \boldsymbol{\Lambda}_k^{(j)} {\boldsymbol{\Lambda}_k^{(j)}}^{-1/2}\Big)}_{=\,\mathrm{o}\;\boldsymbol{I}=\boldsymbol{0}} \\ &\qquad - \eta_{\boldsymbol{\Lambda}'} \mathrm{sym}\Big(\frac{\partial J(\boldsymbol{\Lambda}(\boldsymbol{\Lambda}'))}{\partial \boldsymbol{\Lambda}'}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}}\Big)\Bigg) \\ &= \exp\left(-\eta_{\boldsymbol{\Lambda}'} \mathrm{sym}\Big(\frac{\partial J(\boldsymbol{\Lambda}(\boldsymbol{\Lambda}'))}{\partial \boldsymbol{\Lambda}'}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}}\Big)\right), \end{aligned} \tag{14}$$

where $\boldsymbol{I}\in\mathbb{R}^{m\times m}$ is an identity matrix and

$$\begin{aligned} \frac{\partial J(\boldsymbol{\Lambda}(\boldsymbol{\Lambda}'))}{\partial \boldsymbol{\Lambda}'} &= \left({\boldsymbol{\Lambda}_k}^{1/2} \left(\frac{\partial J(\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}}\right)^\top {\boldsymbol{\Lambda}_k}^{1/2}\right)^\top \\ &= {\boldsymbol{\Lambda}_k}^{1/2} \frac{\partial J(\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}} {\boldsymbol{\Lambda}_k}^{1/2}. \end{aligned} \tag{15}$$

Thus, we get the update rule for $\boldsymbol{\Lambda}$ as:

$$\begin{aligned} \boldsymbol{\Lambda}_{k+1}^{(j)} &= {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} \boldsymbol{\Lambda}_{k+1}^{\prime(j)} {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} \\ &= {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} \exp\Bigg(\\ &\qquad -\eta_{\boldsymbol{\Lambda}'} {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} \mathrm{sym}\left(\frac{\partial J(\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}}\right) {\boldsymbol{\Lambda}_k^{(j)}}^{1/2} \\ &\Bigg) {\boldsymbol{\Lambda}_k^{(j)}}^{1/2}, \end{aligned} \tag{16}$$

where

$$\begin{aligned} &\frac{\partial J(\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}}\bigg|_{\boldsymbol{\Lambda}=\boldsymbol{\Lambda}_k^{(j)}} \\ &= \sum_{i=0}^{N-1} \Bigg\{ \Big[y^{(i)} - \phi\big(f(\boldsymbol{x}^{(i)})\big)\Big] h_k^{(j)} \mathcal{K}(\boldsymbol{x}^{(i)}, \boldsymbol{c}_k^{(j)}; \boldsymbol{\Lambda}_k^{(j)}) \\ &\qquad (\boldsymbol{x}^{(i)} - \boldsymbol{c}_k^{(j)})(\boldsymbol{x}^{(i)} - \boldsymbol{c}_k^{(j)})^\top \Bigg\}. \end{aligned} \tag{17}$$

B. Updating the kernel means

For the updating of the kernel mean, we apply the steepest descent (SD) method [20] so that the cost function Eq. (7) is minimized. Therefore, the update rule can be obtained as:

$$\boldsymbol{c}_{k+1}^{(j)} = \boldsymbol{c}_k^{(j)} - \eta_{\boldsymbol{c}} \frac{\partial J(\boldsymbol{c})}{\partial \boldsymbol{c}}\bigg|_{\boldsymbol{c}=\boldsymbol{c}_k^{(j)}}, \tag{18}$$

where

$$\begin{aligned} &\frac{\partial J(\boldsymbol{c})}{\partial \boldsymbol{c}}\bigg|_{\boldsymbol{c}=\boldsymbol{c}_k^{(j)}} \\ &= \frac{2}{N} \sum_{i=0}^{N-1} \Bigg\{ \Big[y^{(i)} - \phi\big(f(\boldsymbol{x}^{(i)})\big)\Big] h_k^{(j)} \mathcal{K}(\boldsymbol{x}^{(i)}, \boldsymbol{c}_k^{(j)}; \boldsymbol{\Lambda}_k^{(j)}) \\ &\qquad \boldsymbol{\Lambda}_k^{(j)} (\boldsymbol{x}^{(i)} - \boldsymbol{c}_k^{(j)}) \Bigg\}, \end{aligned} \tag{19}$$

and $\eta_{\boldsymbol{c}}$ is a step for $\boldsymbol{c}$.

C. Fitting the coefficients combined with the kernel parameters updating

As the first step of the model fitting, the support vectors of the proposed method are initialized by using the training set as:

$$\mathcal{K}\Big(\cdot, \boldsymbol{c}_0^{(j)}; \boldsymbol{\Lambda}_0^{(j)}\Big)_{j\in\mathcal{J}_0} = \mathcal{K}\Big(\cdot, \boldsymbol{x}^{(j)}; \boldsymbol{I}\Big)_{j\in\mathcal{J}_0}, \tag{20}$$

Algorithm 1 Model fitting of the KLR-SPD

Input: Training set $\{(\boldsymbol{x}^{(i)},y^{(i)})\}_{i\in\{0,1,\cdots,N-1\}}$
// definition
ℓ_1-regularization parameter λ;
Learning rate for coefficient η_h;
Learning rate for kernel precision $\eta_{\boldsymbol{\Lambda}}$;
Learning rate for kernel mean $\eta_{\boldsymbol{c}}$;
Number of maximum learning epoch k_{max};
// Initialization
$\mathcal{J}_0\leftarrow\{0,1,\cdots,N-1\}$;
$\left\{\boldsymbol{c}_0^{(j)}\right\}_{j\in\mathcal{J}_0}\leftarrow\{\boldsymbol{x}^{(i)}\}_{i=\{0,1,\cdots,N-1\}}$;
$\left\{\boldsymbol{\Lambda}_0^{(j)}\right\}_{j\in\mathcal{J}_0}\leftarrow\{\boldsymbol{I},\cdots,\boldsymbol{I}\}$;
// Fitting
$k\leftarrow0$
while $k\neq k_{max}$ **do**
  Update $\left\{\boldsymbol{\Lambda}_k^{(j)}\right\}_{j\in\mathcal{J}_k}$ to $\left\{\boldsymbol{\Lambda}_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$ by (16);
  Update $\left\{\boldsymbol{c}_k^{(j)}\right\}_{j\in\mathcal{J}_k}$ to $\left\{\boldsymbol{c}_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$ by (18);
  Update $\left\{h_k^{(j)}\right\}_{j\in\mathcal{J}_k}$ to $\left\{h_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$ by (21);
  $\mathcal{J}_{k+1}\leftarrow\{\}$;
  for $j^*\in\mathcal{J}_k$ **do**
    if $h_{k+1}^{(j^*)}=0$ **then**
      Remove $\boldsymbol{\Lambda}_{k+1}^{(j^*)}$ from $\left\{\boldsymbol{\Lambda}_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$;
      Remove $\boldsymbol{c}_{k+1}^{(j^*)}$ from $\left\{\boldsymbol{c}_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$;
      Remove $h_{k+1}^{(j^*)}$ from $\left\{h_{k+1}^{(j)}\right\}_{j\in\mathcal{J}_k}$;
    else
      $\mathcal{J}_{k+1}\leftarrow\mathcal{J}_{k+1}\cup\{|\mathcal{J}_{k+1}|\}$;
    end if
  end for
  $k\leftarrow k+1$;
end while
Output: Support vectors $\{\mathcal{K}(\cdot,\boldsymbol{c}^{(j)};\boldsymbol{\Lambda}^{(j)})\}_{j\in\mathcal{J}_{k-1}}$ and corresponding coefficients $\{h^{(j)}\}_{j\in\mathcal{J}_{k-1}}$

where

$$\mathcal{J}_0=\{0,1,...,N-1\}.$$

Then, the kernel coefficient update method with the ℓ_1-regularization described in II-B is applied as follows:

$$h_{k+1}^{(j)}=\mathrm{sign}\left(\alpha_k^{(j)}\right)\left[\left|\alpha_k^{(j)}\right|-\lambda\eta_h\right]_+, \tag{21}$$

TABLE II
THE LIST OF DATASETS FOR THE EXPERIMENT

Dataset	features	samples	ratio of labels
Australian Credit Approval	14	690	383:307
Breast Cancer Wisconsin	9	683	444:239
Climate Model Simulation Crashes	18	540	46:494
Cryotherapy [21], [22]	6	90	42:48
Diabetic Retinopathy Debrecen	19	1151	540:611
Fertility (fertility)	9	100	88:12:00
German Credit Data	24	1000	700:300
Haberman's Survival	3	306	225:81
Heart	13	270	150:120
Immunotherapy [21], [22]	7	90	19:71
MONK's-1	6	432	216:216
MONK's-2	6	432	290:142
MONK's-3	6	432	204:228
Parkinsons	22	195	48:147
Sonar, Mines vs. Rocks	60	208	97:111
SPECTF Heart	44	267	55:213

where

$$\begin{aligned}\alpha_k^{(j)}&=h_k^{(j)}-\eta_h\left.\frac{\partial J(h)}{\partial h}\right|_{h=h_k^{(j)}}\\&=h_k^{(j)}-\eta_h\frac{1}{N}\sum_{i=0}^{N-1}\left[y^{(i)}-\phi\left(f\left(\boldsymbol{x}^{(i)}\right)\right)\right]\mathcal{K}\left(\boldsymbol{x}^{(i)},\boldsymbol{c}_k^{(j)};\boldsymbol{\Lambda}_k^{(j)}\right).\end{aligned}$$

At this update step, the update methods for the parameters described in III-A and III-B are combined into the fitting of KLR. By using the ℓ_1-regularization and the optimization of the kernel parameters, the number of support vectors can be reduced. The KLR which is applied these update methods is named *KLR-SPD*. Algorithm 1 describes the pseudocode of KLR-SPD.

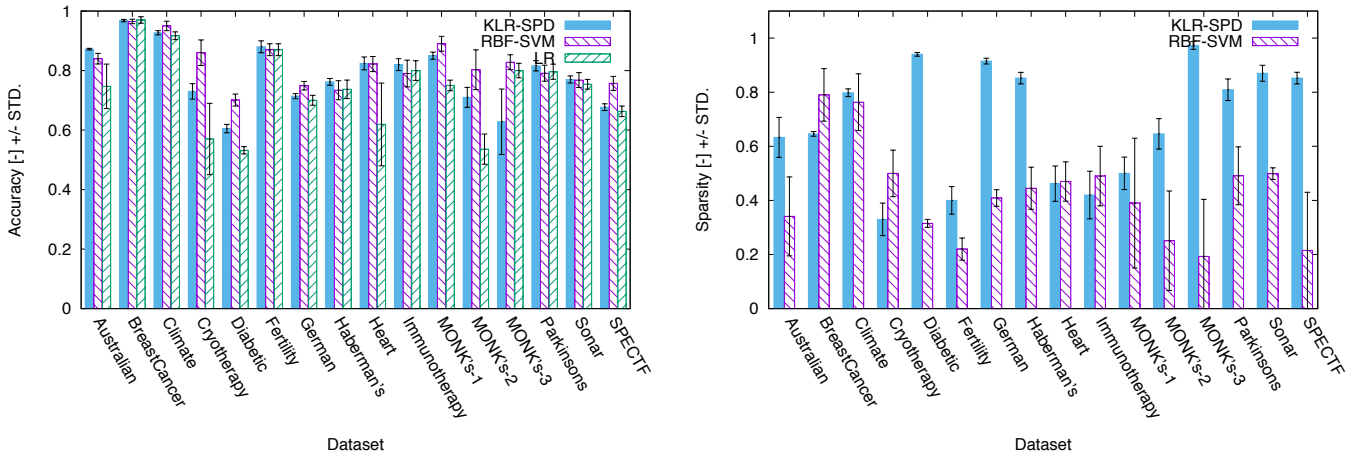
IV. Numerical Experiments

We compare the proposed KLR-SPD, the RBF-SVM [6] and the LR [23] with employ an ℓ_2-regularization. In this numerical experiment, 16 datasets published on UCI Machine Learning Repository [12] are employed to perform the binary classifications. Each dataset is randomly divided into two equal sets, the one of them is used for model fitting as a trainset and the rest is used for model evaluation as a testset. At the model fitting of RBF-SVM or LR, the Gaussian kernel parameter γ and the trade-off parameter C in the RBF-SVM and the ℓ_2-regularization parameter in LR are respectively choosed over the range of $\{0.0001,0.001,0.01,0.1,1,10\}$ by grid search. For adjusting the grid search, the five-fold cross validation with two subsets is used. One subset is used for validation and the other subset is used for model training. On the other hand, the proposed KLR-SPD does not require semi-manual parameter tuning such as grid search.

For the evaluation, we adopt a mean accuracy and a mean sparsity by taking an average over 10 independent realizations.

TABLE III
ACCURACIES AND SPARSITIES OF (MEAN ± STD.) OF EACH COMPARED MODEL. THE BEST ACCURACIES AND SPARSITIES ARE HIGHLIGHTED.

Dataset	Accuracy ± STD.			Sparsity ± STD.	
	KLR-SPD	RBF-SVM	LR	KLR-SPD	RBF-SVM
Australian Credit Approval	**0.872±0.00284**	0.840±0.0179	0.747±0.0746	**0.633±0.0739**	0.341±0.146
Breast Cancer Wisconsin	0.968±0.00388	0.965±0.00745	**0.970±0.0108**	0.646±0.00858	**0.790±0.0972**
Climate Model Simulation Crashes	0.927±0.00718	**0.950±0.0159**	0.917±0.0131	**0.798±0.0144**	0.763±0.105
Cryotherapy	0.73±0.026	**0.86±0.043**	0.57±0.12	0.33±0.060	**0.50±0.086**
Diabetic Retinopathy Debrecen	0.605±0.0141	**0.701±0.0202**	0.532±0.0122	**0.940±0.00701**	0.315±0.0147
Fertility	**0.88±0.020**	0.87±0.020	0.87±0.020	**0.40±0.051**	0.22±0.041
German Credit Data	0.714±0.00849	**0.749±0.0145**	0.700±0.0166	**0.915±0.0107**	0.409±0.0306
Haberman's Survival	**0.762±0.0114**	0.734±0.0319	0.737±0.0312	**0.852±0.0213**	0.445±0.0780
Heart	**0.824±0.0217**	0.822±0.0254	0.619±0.139	0.462±0.0654	**0.470±0.0727**
Immunotherapy	**0.82±0.020**	0.79±0.045	0.80±0.033	0.42±0.088	**0.49±0.11**
MONK's-1	0.85±0.012	**0.89±0.025**	0.75±0.018	**0.50±0.060**	0.39±0.24
MONK's-2	0.710±0.0335	**0.803±0.0669**	0.536±0.0509	**0.646±0.0565**	0.251±0.184
MONK's-3	0.628±0.110	**0.828±0.0251**	0.800±0.0245	**0.973±0.0147**	0.193±0.211
Parkinsons	**0.816±0.0174**	0.791±0.0269	0.796±0.0254	**0.809±0.0400**	0.491±0.107
Sonar, Mines vs. Rocks	**0.770±0.0119**	0.768±0.0251	0.754±0.0162	**0.870±0.0295**	0.499±0.0214
SPECTF Heart	0.677±0.0115	**0.757±0.0235**	0.663±0.0178	**0.852±0.0217**	0.215±0.215



(a) Accuracies (b) Sparsities

Fig. 1. Accuracies and sparsities shown in Table III are displayed as bar graphs. And STDs are indicated by error bars.

The accuracy and the sparsity are calculated by:

$$\text{Accuracy}=\frac{n_{\text{correct}}}{N_{\text{test}}}, \tag{22}$$

$$\text{Sparsity}=1-\frac{n_{\text{sup}}}{N}, \tag{23}$$

where n_{correct}, N_{test} and n_{sup} are a number of correct predictions, a number of test set and a number of support vectors, respectively.

The results of experiment are shown in Table III and Fig. 1. From Table III and Fig. 1(a), it can be seen that the accuracies of the KLR-SPD achieved almost comparable accuracies to the RBF-SVM, which had parameter tuning using grid search. However, in most cases, the sparsities of the KLR-SPD are higher than the other mothods as confirmed in Table III and Fig. 1(b). In other words, the KLR-SPD showed almost the same classification performance as RBF-SVM with a small number of support vectors.

V. CONCLUSION

We proposed a flexible fitting method that optimize the kernel parameters to apply the binary classifier. The our proposed method is applied to kernel logistic regression, then the model parameters include kernel parameters are fully data-driven. Especially, thanks to employ the generalized Gaussian kernel and its data-driven optimization, it can be conceivable that the feature vectors are projected flexibly. In addition, it is possible to constitute a sparse model by using the ℓ_1-regularization. The proposed method demonstrated its effectiveness by the numerical experiment.

ACKNOWLEDGMENT

The authors would like to thank Dr. Simone Fiori for fruitful discussion on Riemannian geometry.

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.

[2] L. Zhou, S. Pan, J. Wang, and A.V. Vasilakos, "Machine learning on big data: Opportunities and challenges," Neurocomputing, vol.237, pp.350–361, 2017.

[3] Q. Zhang, L.T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," Information Fusion, vol.42, pp.146–157, 2018.

[4] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "GAN-based synthetic medical image augmentation for increased cnn performance in liver lesion classification," Neurocomputing, vol.321, pp.321–331, 2018.

[5] Y. Luo and B.-L. Lu, "EEG data augmentation for emotion recognition using a conditional wasserstein GAN," Proc. of 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), vol.2018, 07 2018. 2535 - 2538.

[6] C.J. Burges, "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery, vol.2, no.2, pp.121–167, 1998.

[7] O.L. Mangasarian, et al., "Generalized support vector machines," Advances in Neural Information Processing Systems, pp.135–146, 1999.

[8] R. Herbrich, Learning Kernel Classifiers: Theory and Algorithms, MIT Press, Cambridge, MA, USA, 2001.

[9] T. Wada, K. Fukumori, and T. Tanaka, "Dictionary learning for Gaussian kernel adaptive filtering with variable kernel center and width," Proc. of 2018 International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018), 04 2018. 2766-2770.

[10] K. Fukumori and T. Tanaka, "Fully data-driven optimization of gaussian parameters for kernel classifier," 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Nov. 2018. 1858-1863.

[11] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," J. Mach. Learn. Res., vol.2, pp.67–93, 2002.

[12] C. Blake and C. Merz, "UCI repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science," 1998.

[13] S.S. Keerthi, K. Duan, S.K. Shevade, and A.N. Poo, "A fast dual algorithm for kernel logistic regression," Machine Learning, vol.61, no.1, pp.151–165, 2005.

[14] F. Liu, X. Huang, and J. Yang, "Indefinite kernel logistic regression," Proceedings of the 2017 ACM on Multimedia ConferenceACM 2017. 846–853.

[15] N. Aronszajn, "Theory of reproducing kernels," Trans. Amer. Math. Soc., vol.68, no.9, pp.337–404, 1950.

[16] B. Pan, W.-S. Chen, B. Chen, C. Xu, and J. Lai, "Out-of-sample extensions for non-parametric kernel methods," IEEE Transactions on neural networks and learning systems, vol.28, no.2, pp.334–345, 2017.

[17] G.C. Cawley and N.L. Talbot, "Efficient model selection for kernel logistic regression," Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, vol.2, 2004. 439–442.

[18] J. Duchi and Y. Singer, "Efficient online and batch learning using forward backward splitting," Journal of Machine Learning Research, vol.10, no.Dec, pp.2899–2934, 2009.

[19] K. Tsuda, G. Rätsch, and M.K. Warmuth, "Matrix exponentiated gradient updates for on-line learning and bregman projection," J. Mach. Learn. Res., vol.6, no.Jun, pp.995–1018, 2005.

[20] S. Haykin, Adaptive Filter Theory, Upper Saddle River, NJ: Prentice-Hall, 2002.

[21] F. Khozeimeh, R. Alizadehsani, M. Roshanzamir, A. Khosravi, P. Layegh, and S. Nahavandi, "An expert system for selecting wart treatment method," Comput. Biol. Med., vol.81, no.C, pp.167–175, Feb. 2017. https://doi.org/10.1016/j.compbiomed.2017.01.001

[22] F. Khozeimeh, F. Jabbari Azad, Y. Mahboubi Oskouei, M. Jafari, S. Tehranian, R. Alizadehsani, and P. Layegh, "Intralesional immunotherapy compared to cryotherapy in the treatment of warts," International Journal of Dermatology, vol.56, no.4, pp.474–478, 2017.

[23] C.M. Bishop, Pattern Recognition and Machine Learning, New York, NY: Springer, 2006.