Journal of Pattern Recognition Research 1 (2010) 38-51

Received July 14, 2009. Revised October 30, 2009. Accepted July 1, 2010.

A Least Square Kernel Machine With Box Constraints

Jayanta Basak*
IBM Research - India
4, Block-C, Institutional Area, Vasant Kunj, New Delhi - 110070, India.

basakjayanta@yahoo.com, basakjayanta@gmail.com

Abstract

Principle of parsimony (Occams razor) is a key principle where the unnecessary complexity of a classier is regulated to improve the generalization performance in pattern classication. In decision tree construction, often the complexity is regu- lated by early stopping the growth of a tree at a node whenever the impurity at that node reaches below a threshold. In this paper, we generalize this heuristic and express this principle in terms of constraining the outcome of a classier instead of explicitly regularizing the model complexity in terms of the model parameters. We construct a classier using this heuristic namely, a least square kernel machine with box constraints (LSKMBC). In our approach, we consider uniform priors and obtain the loss functional for a given margin considered to be a model selection parameter. The framework not only di?ers from the existing least square kernel machines, but also it does not require Mercer condition satisability. We also discuss the relation- ship of the proposed kernel machine with several other existing kernel machines. Experimentally we validate the performance of the classier over real-life datasets, and observe that LSKMBC performs competitively, and is able to produce certain results even better than SVM.

WWW.JPRR.ORG

1. Introduction

In support vector machines (SVM) [25], the margin between two classes is maximized in a higher dimensional space $\phi(.)$ under the constraint of inequality type $t_i(w^T\phi(x_i)+b) \geq 1$ where w is the separating hyperplane. The problem is transformed into an unconstrained problem by the method of Lagrange undetermined multipliers such that the functional

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i(t_i(w^T\phi(x_i)+b)-1) \tag{1}$$

is minimized with respect to $\mathbf{w}$, b, and maximized with respect to $\alpha_i \geq 0$. Finally introducing slack variables for non-separable cases, and taking into account of the inner product in the Hilbert space, the functional $W_{svm}(\alpha)$ in the dual space is expressed as

$$W_{svm}(\alpha) = \sum_i \alpha_i - \frac{1}{2C}\sum_{i,j}\alpha_i\alpha_j t_i t_j K(x_i,x_j) \tag{2}$$

subject to $0 \leq \alpha_i \leq 1$ and $\sum_i \alpha_i t_i = 0$, where C is judiciously chosen constant and $K(x_i,x_j) = \phi(x_i)\phi(x_j)$ is a symmetric kernel expressible as an inner product in the higher dimensional space subject to Mercer condition [25]. Once the functional $W_{svm}(\alpha)$ is maximized with respect to α_is (support vectors), classlabel of a test sample is obtained as $sign(\sum_i \alpha_i K(x,x_i)t_i + b)$. Support vector machines have been generalized to multiclass classification, and also used in one-against-all classification.

In the least square kernel machines, the quadratic optimization functional is replaced by linear functional where priors over the Lagrangians are subjected to spherical Gaussian

* The author is presently affilated with NetApp Advanced Technology Group, Bangalore, India.

© 2010 JPRR. All rights reserved. Permissions to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or to republish, requires a fee and/or special permission from JPRR.

distribution. For example, in the ridge regression framework [9], the least square regression includes a regularization factor such that

$$L = \|t - w^T x\|^2 + a\|w\|^2 \tag{3}$$

where a is a Lagrangian. The ridge regression framework introduces priors over the coefficients w for a smooth regression function estimation, the priors being subjected to a spherical Gaussian prior distribution. In adaptive ridge regression [17], automatic relevance determination is performed in such a way that each variable is penalized by the method of automatic balancing while keeping the average penalty constant. In RLSC [15], the objective functional is derived as a simple square-loss function and regularized using the Tikhonov regularization [22] with a quadratic term in α vectors. In RLSC reproducing Hilbert kernels are considered such that the objective functional takes a form

$$L = \frac{1}{N}(t - K\alpha)^T(t - K\alpha) + \frac{\gamma}{2}\alpha^T K\alpha \tag{4}$$

where N is the number of training samples, γ is a regularization parameter, and α are the coefficients (the notation of α is same throughout this article). This convex optimization problem is then transformed into a set of linear equations (similar to least square regression) by equating the first order derivative to zero based on the assumption that the underlying noise is generated by a Gaussian model. The coefficients are then derived from the resulting set of linear equations such that

$$(K + \gamma NI)\alpha = t \tag{5}$$

In least-square support vector machine (LSSVM) [20, 24], a squared error term is considered in addition to a regularization term $w^T w$ where w represents the directionality of the separating hyperplane. The squared error is derived based on the equality constraints instead of the inequality constraints as used in SVM. The convex optimization functional is then transformed into a set of linear equations by equating the first derivative to zero using unconstrained priors α.

Constrained parameters have been used in LASSO (least absolute shrinkage and selection operator) [21, 11], and generalized LASSO [16, 8]. In LASSO, the loss functional considered is given as

$$L = \|t - K\alpha\|_2^2 \tag{6}$$

subject to $\|\alpha\|_1 < \lambda$ where λ is a constant which determines the approximation accuracy. Our model is similar to the LASSO as interpreted in Equation (23) except that the constraints over the parameters are different. Considering the exponential hyperpriors, the LASSO model can be expressed in the form

$$L = \|t - K\alpha\|_2^2 + b\|\alpha\|_1 \tag{7}$$

where b is a Lagrange multiplier determined by the exponential hyperpriors. The generalized LASSO [16] uses a very similar form of loss functional as in Equation (7) except that it uses a more robust form of Huber loss [19] which is quadratic for smaller deviation and linear for larger deviations. Subsequently iterative reweighted least square (IRLS) technique is used to optimize the functional. The constraints that we use are different from the constraints used in LASSO and generalized LASSO.

In relevance vector machines [23], empirical error is defined based on the logistic regression model where the outcome is given as $g(x) = 1/(1+\exp(-w^T\phi(x))$. In a Bayesian framework

zero-mean Gaussian priors are defined over the parameter vectors w such that $p(w|\alpha) = \prod_i \mathcal{N}(0, 1/\alpha_i)$ where $\mathcal{N}$ represents the Gaussian distribution. In this framework, an iterative method is employed. It starts with a current estimate of α vectors, and based on this estimate, the most probable hyperprior (prior variance) parameters are chosen. The prior variance parameters are then used in the posterior to get new estimates of α. Thus relevance vector machine explicitly regularizes the model parameters w within a Bayesian framework with respect to certain priors and iteratively estimates the priors. We do not use any logistic regression in a Bayesian framework.

In this paper, we propose a least square kernel machine with box constraitnts. In the proposed variant of the least square kernel machine, we do not consider Gaussian hyperpriors and therefore we do not consider linear regression to determine the model coefficients. Rather, we formulate the machine with uniform hyperpriors and constrain the model coefficients within a box. Experimentally, we show that for certain datasets, the proposed kernel classifier is able to outperform the SVM as well as LSSVM.

2. Least Square Kernel Machine With Box Constraint

We derive a least-square kernel machine by reducing the error over training samples and simultaneously reduce the excess outcome by comparing it with a given threshold. The outcome of the least square kernel machine is

$$g(x, \alpha) = \sum \alpha_i K(x, x_i) t_i \tag{8}$$

where $\alpha_i \in [0, 1]$ is a weight for the training sample x_i, $K(.,.)$ is a kernel function, and t_i is the classlabel associated with the sample x_i.

2.1 Proposed Kernel Machine

We formulate the classifier for two-class classification task and then generalize it for multi-class classification using one-against-all strategy. For a two-class classifier with known sample labels $t \in \{-1, 1\}$, we formulate the loss function as

$$L = \frac{1}{2} \sum_i (\sum_j \alpha_j K(x_i, x_j) t_j - \lambda t_i)^2 \tag{9}$$

where $\lambda \in (0, 1]$ is a given threshold acting as a model selection parameter. For a given λ, the minimization of L is equivalent to maximization of the functional

$$W(\alpha) = \lambda t^T K D(t) \alpha - \frac{1}{2} \alpha^T D(t) K^T K D(t) \alpha \tag{10}$$

subject to $0 \leq \alpha_i \leq 1$ for all i. $D(t)$ is a diagonal matrix with the diagonal equal to t.

Once we obtain the pattern vectors with non-zero α values by maximizing the functional $W(\alpha)$ with respect to α_is (Equation 10), we classify any new test sample x, i.e., assign a classlabel t to x as

$$t = \begin{cases} 1 & \text{if } \sum_i \alpha_i K(x, x_i) t_i \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{11}$$

Multi-class Classification: In the case of multi-class classification, we consider the one-against-all strategy [14], i.e., classify the patterns of each class against all other classes and obtain the coefficients for each class separately. Formally let there be l classlabels

$L = \{1, 2, \cdots, l\}$. Each time we consider one classlabel at a time. Let the classlabel of concern be c. In that case $t_i \in L$ is transformed into a vector $\{t_{i1}, t_{i2}, \cdots, t_{il}\}$ such that

$$t_{ic} = \begin{cases} 1 & \text{if } t_i = c \\ -1 & \text{otherwise} \end{cases} \tag{12}$$

For each classlabel c, we compute the α_{ic}s separately, and the classlabel t to a new sample x is assigned as

$$t = argmax_{c \in L}\{\sum_i \alpha_{ic} K(x, x_i) t_{ic}\} \tag{13}$$

where α_{ic}s are the non-zero coefficients.

2.2 Interpretation of the Kernel Machine

The total normalized loss over the training dataset $\mathcal{D}$ is

$$L(\mathcal{D}|\alpha) = \frac{1}{2|\mathcal{D}|} \sum_{x \in \mathcal{D}} (g(x, \alpha) - \lambda t(x))^2 \tag{14}$$

The Equation (14) can be expressed as

$$L(\mathcal{D}|\alpha) = \frac{1}{2|\mathcal{D}|} \sum_x [\lambda(g(x, \alpha) - t(x))^2 + (1 - \lambda) g^2(x, \alpha) - \lambda(1 - \lambda) t^2(x)] \tag{15}$$

Since the third term in Equation (15) is independent of α, we can equivalently express $L(\mathcal{D}|\alpha)$ as

$$L(\mathcal{D}|\alpha) = \frac{1}{2|\mathcal{D}|} \left[\sum_x (g(x, \alpha) - t(x))^2 + \frac{1 - \lambda}{\lambda} \sum_x g^2(x, \alpha) \right] \tag{16}$$

The second term in Equation (16) imposes explicit regularization on the outcome of the classifier and $\frac{1-\lambda}{\lambda}$ is the Lagrangian multiplier of the regularization functional. Since $\sum_{x \in \mathcal{D}} g^2(x, \alpha)$ is a function of α only, we can equivalently express Equation (16) as

$$L(\mathcal{D}|\alpha) = \frac{1}{2|\mathcal{D}|} \left[\sum_x (g(x, \alpha) - t(x))^2 + \gamma \Gamma(\alpha) \right] \tag{17}$$

where $\gamma = (1 - \lambda)/\lambda$ is the Lagrangian multiplier and

$$\Gamma(\alpha) = \sum_{x \in \mathcal{D}} g^2(x, \alpha) \tag{18}$$

is a function of the model parameters α; $\Gamma(\alpha)$ represents a form of model complexity dependent on the outcome of the classifier. Thus although we reduce the superfluity in the outcome, it is connected to the explicit regularization of the model complexity. We observe that the contribution of the regularization term increases as we decrease λ and effectively bias increases. On the other hand for $\lambda = 1$, the regularization term vanishes.

From Equations (8) and (18),

$$\Gamma(\alpha) = \alpha^T D(t) K^T K D(t) \alpha \tag{19}$$

Alternatively, from Equations (17) and (19), the normalized loss function can be expressed as

$$L = \frac{1}{2|\mathcal{D}|} \left[(t - K\beta)^T (t - K\beta) + \gamma \beta^T K^T K \beta \right] \tag{20}$$

where $\beta \in [-1, 1]$ and $\beta_i t_i \geq 0$. We express the objective functional in this form in order to show the similarity with other existing kernel machines in the next section.

As we impose the constraints on the values of α, the loss can be reformulated as

$$L = \frac{\lambda^2}{2} \sum_i (\sum_j \frac{\alpha_j}{\lambda} K(x_i, x_j) t_j - t_i)^2 \tag{21}$$

which is equivalent to minimizing a functional

$$L = \frac{1}{2} \sum_i (\sum_j K(x_i, x_j) \hat{\alpha}_j t_j - t_i)^2 \tag{22}$$

subject to the constraints $0 \leq \hat{\alpha}_i \leq \frac{1}{\lambda}$. If we replace $\hat{\alpha}_i t_i$ by $\hat{\beta}_i$ then we obtain the quadratic loss

$$L = \frac{1}{2} \sum_i (\sum_j K(x_i, x_j) \beta_j - t_i)^2 \tag{23}$$

subject to the constraint that $\hat{\beta} \in [-\frac{1}{\lambda}, +\frac{1}{\lambda}]$ and $\hat{\beta}_i t_i \geq 0$. In other words, we minimize a quadratic loss functional equivalent to the empirical error subject to constraining the model coefficients in box; the size of the box depends on the parameter λ.

We also observe that the classifier has a connection to the weighted Parzen window set-up [1]. We find out the weights or the model coefficients by minimizing the least square formulation which minimizes the difference between the outcome of the classifier and the given threshold. The formulation is equivalent to minimizing the empirical error on the training data subject to constraining the model parameters within a box, the size of the box being determined by the threshold. We call this specific kernel machine as a least-square kernel machine with box constraints (LSKMBC). LSKMBC is derived purely from the regularization perspective and not from the margin maximization perspective, and therefore it is not a variant of SVM. Since we use a weighted kernel set-up such that estimated density matches with the training distribution, we are not restricted to using Mercer kernels; Mercer kernels are essentially expressible in terms of the dot product of two vectors in certain higher dimensional space as used in SVM and their variants. In the optimization function as in Equation (10), the quadratic term is always positive semi-definite irrespective of whether we use Mercer or non-Mercer kernels.

2.3 Relationship with Other Kernel Machines

The effects of αs are similar to that used in the support vector machines (SVM) [6, 25] where αs define the support vectors. However the quadratic function to be maximized in SVM is derived on the basis of margin maximization with respect to optimal separating hyperplane in some higher dimensional space $\phi(.)$ such that K is expressible as a inner product, $K(x, x_i) = \phi(x)\phi(x_i)$, and αs are the Lagrangians. In our case, we do not explicitly consider the margin maximization with respect to separating hyperplane in the space of $\phi(.)$, rather we minimize the outcome for a given parameter λ.

This leads to a difference in the quadratic term $K^T K$ in our formulation in Equation (10) from that in the SVM. Apart from the quadratic term, we also observe the difference with SVM in the linear term. In SVM, due to margin maximization all α_is contribute equally in the linear term as in

$$W_{svm}(\alpha) = \mathbf{1}^T \alpha - \frac{1}{2C} \alpha^T D(t) K D(t) \alpha \tag{24}$$

with an additional constraint that $\alpha^T t = 0$ and $\alpha_i \geq 0$ for all i. In the modified framework of soft margin SVM [18, 2], the funcitonal is expressed as

$$W_{softsvm}(\alpha) = \mathbf{1}^T\alpha - \frac{1}{2}\alpha^T D(t)(K + \frac{1}{C}I)D(t)\alpha \tag{25}$$

where I is the identity matrix with the same constraint i.e., $\alpha^T t = 0$. We do not require this constraint (Equation (10)) since different α_is contribute differently in the linear term depending on the kernel function and the distribution of the patterns.

As we view the expanded quadratic loss in Equation (10), we observe the similarity of LSKMBC with ν-SVM [5], where the optimization functional is given as

$$L = -\frac{1}{2}\alpha^T D(t)KD(t)\alpha \tag{26}$$

subject to $0 \leq \alpha_i \leq \frac{1}{\lambda}$, λ being a regularization parameter $\lambda > 0$, and $\alpha^T t = 0$, $\sum \alpha_i \geq \nu$. In ν-SVM the quadratic optimization in ν-SVM does not contain any linear term as in standard SVM. The performance of ν-SVM [5] depends on two parameters namely ν and λ. ν-SVM uses a principle of regularizing the margin of separation between the classes (with an additional regularization factor) where the parameter ν controls the penalization due to separation of the classes in the primal formulation of the loss functional. The separation between the classes is governed by the separation margin ρ and a regularization functional $-\nu\rho$ is added to the primal formulation. The minimization of the loss functional is subjected to the maximization of the separation margin ρ and the effect of ρ depends on the parameter ν. In this respect, LSKMBC also uses a parameter λ to control the separation between the classes in terms of the outcome of the classifier. However, the primal formulation of ν-SVM is derived from the margin maximization principle as in the SVM with an additional regularization factor $\nu\rho$ and therefore uses Mercer kernels.

We observe that our model as expressed in Equation (20) is similar to the adaptive ridge regression (AdR) and RLSC. However, in AdR, RLSC, and LSSVM, the form of model complexity that has been used to regularize the model parameters is different from that in the LSKMBC. Therefore minimization of L with unconstrained parameters (α) in such models does not yield simple scaling of the outcome $g(x, \alpha)$ with the Lagrangian parameter. In our model on the other hand, unconstrained minimization results into simple scaling with respect to λ, and we constrain the parameters α within a box. Moreover, RLSC and other related family of kernel machines are restricted to Hilbert kernels subjected to Mercer condition which is not true in our model.

We observe the similarity with RLSC and LSSVM in the formulation except that both RLSC and LSSVM assume Gaussian hyperpriors and employ Tikhonov regularization. On the other hand, LSKMBC do not employ Tikhonov regularization, however, deviates from the assumption of Gaussian hyperpriors and employ uniform hyperpriors with box constraint derived from a given margin λ. Also LSKMBC does not require the Mercer condition satisfiability of the kernels which is required in RLSC and SVM. Generalized LASSO also deviates from the Gaussian hyperprior model and develops on a more robust Huber loss measure. LSKMBC also has a similarity with ν-SVM [5], where the optimization functional is given as

$$L = -\frac{1}{2}\alpha^T D(t)KD(t)\alpha \tag{27}$$

subject to $0 \leq \alpha_i \leq \frac{1}{\lambda}$, λ being a regularization parameter $\lambda \in [0, 1]$, and $\alpha^T t = 0$, $\sum \alpha_i \geq \nu$. However, ν-SVM does not contain any linear term and it is not derived based on uniform hyperpriors with box constraints.

3. Experimental Results

3.1 Nature of the LSKMBC

We first illustrate the behvior of the LSKMBC with Gaussian kernel given as

$$K(x, x_j) = \exp(-\frac{\|x - x_j\|^2}{2\sigma^2}) \tag{28}$$

where the parameter σ decides the width of the kernel.

In Figure 1, we illustrate the non-zero vectors generated by LSKMBC for two-dimensional Gaussian parity problem. The non-zero vectors are marked by 'o', and the samples from two different classes are marked by 'x' and '.' respectively. We observe that even though we do not perform margin maximization as performed in SVM, the non-zero vectors concentrate towards the boundary between opposite classes. However, this behavior of the LSKMBC is specific to the choice of kernel. Figure 2 illustrates the change in the non-zero vectors as we change σ of Gaussian kernel as 0.2, 0.5, 1 and 2. We observe that for a low σ, the non-zero vectors are not necessarily concentrated near the opposite classes. This is due to the fact that we do not use the margin maximization principle, rather we find non-zero vectors such that existence of other points becomes interpretable subject to a given threshold. As we increase σ, the non-zero vectors start getting concentrated towards the opposite class samples. This is due to the higher interaction between samples from other classes caused by larger width of the kernel function. In Figure 3, we observe that the number of the non-zero vectors decreases with the decrease in λ and vice-versa. This is due to the fact that if we reduce the threshold λ, we require less number of non-zero vectors to support the existence of other samples from the same class subject to the threshold.

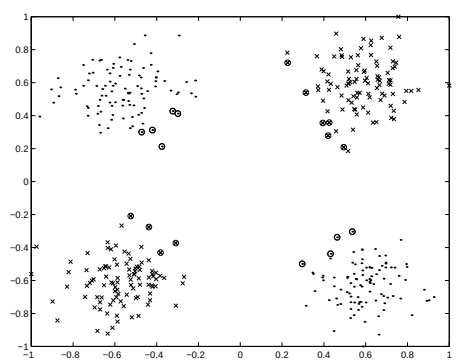

Fig. 1: Non-zero vectors obtained by the LSKMBC for a two-dimensional Gaussian parity problem. The two classes are represented by 'x' and '.' respectively. The non-zero vectors are marked by 'o'. The Gaussian kernel of the LSKMBC is chosen such that $2\sigma^2 = 1$.

3.2 Performance of the LSKMBC

We demonstrate the effectiveness of our classifier on certain real-life data sets as available in the UCI machine learning repository [3]. We normalize all input patterns such that any component of $\mathbf{x}$ lies in the range $[-1, +1]$.

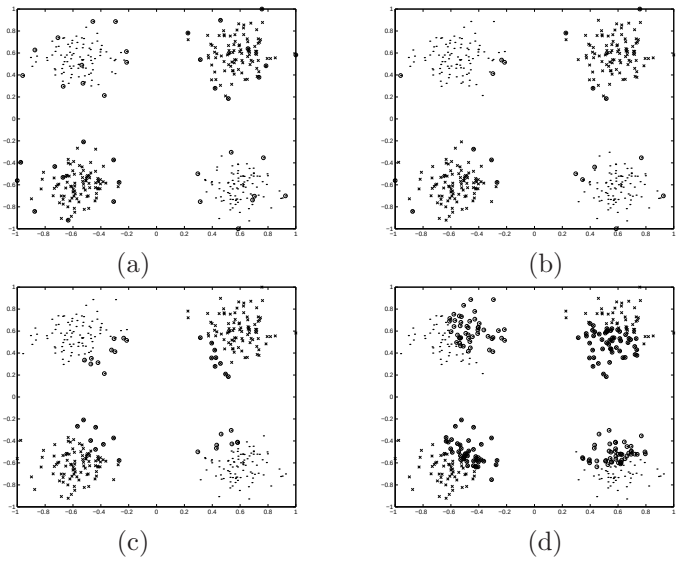

Fig. 2: Non-zero vectors obtained by LSKMBC for the same two-dimensional Gaussian parity problem as in Figure 2 with different kernel sizes. (a), (b), (c), and (d) illustrate the non-zero vectors for $\sigma = 0.2$, $\sigma = 0.5$, $\sigma = 1.0$ and $\sigma = 2.0$ respectively.

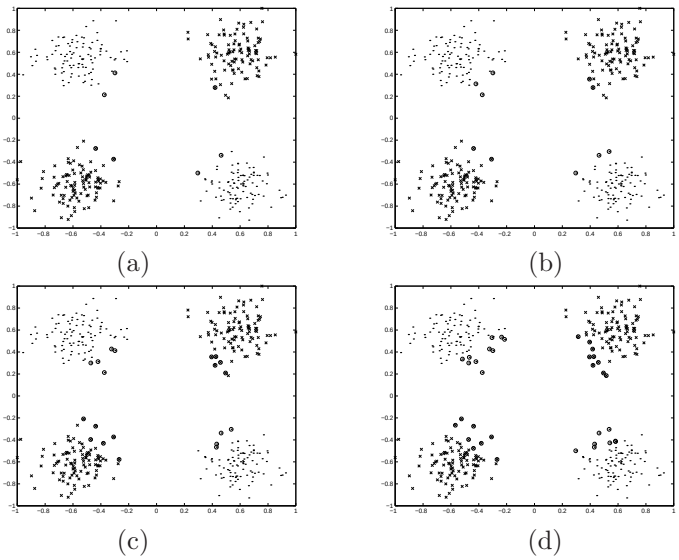

Fig. 3: Non-zero vectors obtained by LSKMBC for the same two-dimensional Gaussian parity problem as in Figure 2 for different margins (λ) with a fixed size of Gaussian kernel $\sigma = 1$. (a), (b), (c), and (d) illustrate the non-zero vectors for $\lambda = 0.1$, $\lambda = 0.2$, $\lambda = 0.5$, and $\lambda = 1.0$ respectively.

Table 1: Description of the pattern classification datasets obtained from the UCI machine learning repository.

Data Set	No. of Instances	No. of Features	No. of Classes
Indian diabetes (Pima)	768	8	2
Diagnostic Breast Cancer (Wdbc)	569	30	2
Prognostic Breat Cancer (Wpbc)	198	32	2
Liver Disease (Bupa)	345	6	2
Flower (Iris)	150	4	3
Bacteria (Ecoli)	326	7	5

3.2.1 Choice of the Kernel Functions

The kernel functions of the LSKMBC is not necessarily subjected to Mercer condition (Hilbert kernels) since the quadratic term in the objective functional of LSKMBC (Equation (10)) is always positive semi-definite. Apart from Gaussian kernels, we select two other kernel functions which are centrally peaked with longer tails such that the distant non-zero vectors have greater interaction between. The first one is given as

$$K_1(x,\mu) = \frac{1}{1+\left(\frac{\|x-\mu\|}{\sigma}\right)^2} \tag{29}$$

The kernel in Equation (29) is derived from the Cauchy distribution which has a long tail, and we call this kernel function as Cauchy kernel. Note that, Cauchy kernel has also been applied in the formation of sparse codes for natural scenes leading to a complete family of localized, oriented, bandpass receptive fields [10].

The second form of kernel is a mixture of Guassian and exponential kernel such that for smaller deviation it is squared loss and for larger deviation the loss is linear (a mixture of quadratic and linear loss has been used in [16, 19], and we borrowed the same idea in designing this kernel function). The second kernel function is given as

$$K_2(x,\mu) = \begin{cases} \exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right) & \text{for } \|x-\mu\| \leq \epsilon \\ \exp\left(-\frac{\|x-\mu\|}{\sqrt{2}\sigma}\right) & \text{otherwise} \end{cases} \tag{30}$$

Note that, K_2 is continuous only for $\epsilon = \sqrt{2}\sigma$ and we choose the same value although in the framework of the LSKMBC it is not necessary that the kernel function needs to be continuous. We refer to the kernel K_2 as the *Gaussian + Exponential* kernel which is not differentiable on the hyperplane $\|x-\mu\| = \sqrt{2}\sigma$. In the next section, we demonstrate the performance of LSKMBC with *Cauchy* kernel and *Gaussian + Exponential* kernel in addition to *Gaussian* kernel.

3.2.2 Results

We report the 10-fold cross-validation performance of the LSKMBC for Gaussian kernel, Cauchy kernel, and Gaussian+Exponential kernel respectively. In computing the 10-fold

cross-validation scores, we randomly partitioned the data into 10 disjoint groups, and used 90% (i.e., 9 disjoint groups) samples as the training sample and the rest 10% samples as the test samples, and iterated this procedure using each group in the partition as the test set and the rest as the training set. We used 10 such trials of random partitioning the data. For each trial, we compute the mean and variance of the classification score over the ten different test sets, and then compute the overall mean and variance scores over the ten different trials. We performed this random partitioning of the dataset and averaged over ten different trials in order to reduce any inherent bias generated due to sample ordering present in the data set. We also compare the performance of the LSKMBC with that of the SVM using Gaussian kernel and third degree polynomial kernels. Note that, it is not possible to use Cauchy and Gaussian+Exponential kernels in SVM since these kernels are not Mercer kernels. In addition we compare the performance with the LSSVM [20] using Gaussian kernels. In comparing the performance, we use the same training set and the test set for each trial and each fold throughout for all these classifiers.

Table 2 summarizes the 10-fold cross-validation score of the LSKMBC on the datasets in Table 1. We report the mean accuracy as well as the standard deviation in the accuracy as the indicator of significance. As a comparison, we also provide the same scores of SVM for Gaussian kernel and polynomial kernel, and LSSVM with Gaussian kernel in Table 2. For implementing SVM, we used the publicly available code in [4]. We implemented the LSSVM using the Matlab code available in [12]. In implementing the LSSVM for multi-class classification (such as 'Iris' and 'Ecoli'), we used 'minimum output coding' [12] which provided the best performance for the LSSVM. In Table 2, we report the best performances of all the classifiers namely, SVM, LSSVM, and LSKMBC and the corresponding parameter values. We observed that all these three classifiers are able to obtain similar scores over a significantly large range of parameter values, and we report one candidate set of values in Table 2. From Table 2, we observe that for the 'Pima', 'Bupa', 'Wpbc', 'Iris' and 'Ecoli' datasets, LSKMBC outperforms SVM and LSSVM, particularly when we observe the performance of the LSKMBC with Cauchy kernel. For the 'Wdbc' dataset, the best performance of SVM is marginally better than that of LSKMBC although LSSVM is much worse than the other two. Interestingly, we observe that the performance of LSKMBC often improves significantly with long-tailed kernels such as Cauchy kernel.

In order to establish the comparison between these classifiers in a more quantitative way, we performed resampled paired t-test as provided in [7]. As described in [7], we randomly divided the dataset such that two-third of the dataset constituted a training set, and the rest one-third constituted a test set. We then used the same training set and test set for all the variants of the three classifiers. We then computed the rate of misclassification on the test set. We repeated this experiment for 30 different trials, where in every trial we randomly partitioned the data and computed the rate of misclassification by each classifier. For every trial, we computed the difference in the rate of misclassification. For example, if c_1 and c_2 are two different classifiers with rates of misclassification $p_1^{(i)}$ and $p_2^{(i)}$ respectively for trial i, then the difference in misclassification is $p^{(i)} = p_1^{(i)} - p_2^{(i)}$, and the statistic is obtained as

$$t = \frac{\bar{p}.\sqrt{N}}{\sqrt{\frac{\sum_{i=1}^{N}(p^{(i)}-\bar{p})^2}{N-1}}} \tag{31}$$

where $N(= 30)$ is the number of trials, and $\bar{p} = \frac{1}{N}\sum_{i=1}^{N} p^{(i)}$ is the average difference in the misclassification rate. Evidently, if $t < 0$ then classifier c_1 is better than the classifier c_2

Table 2: Classification performance in terms of the 10-fold cross-validation scores on the datasets described in Table 1. Classification performance summarizes the best mean classification accuracy and the corresponding standard deviation of the accuracies as a significance check. Corresponding to each classifier model, the candidate parameter values for which the best performance is obtained, are shown.

Classifier	Pima	Bupa	Wdbc	Wpbc	Iris	Ecoli
SVM	76.88	69.39	98.03	80.78	96.13	88.17
(Gaussian)	($\pm$4.74)	($\pm$6.53)	($\pm$1.80)	($\pm$6.15)	($\pm$4.87)	($\pm$4.53)
(σ, C)	$(1, 2)$	$(0.3, 1)$	$(2, 20)$	$(1, 2)$	$(1, 1)$	$(0.5, 1)$
SVM	75.58	71.72	96.20	74.39	96.13	87.25
(Polynomial)	($\pm$4.74)	($\pm$6.40)	($\pm$2.26)	($\pm$8.72)	($\pm$5.06)	($\pm$4.88)
(C)	(1)	(5)	(1)	(1)	(5)	(3)
LSSVM	76.28	68.87	93.78	78.13	84.53	77.40
(Gaussian)	($\pm$5.18)	($\pm$7.21)	($\pm$2.93)	($\pm$3.93)	($\pm$7.90)	($\pm$5.54)
(σ^2, γ)	$(5, 0.1)$	$(1, 2)$	$(4, 0.3)$	$(5, 4)$	$(0.2, 15)$	$(1, 10)$
LSKMBC	77.51	72.42	97.68	81.05	95.87	87.93
(Gaussian)	($\pm$5.41)	($\pm$6.37)	($\pm$1.65)	($\pm$7.11)	($\pm$5.02)	($\pm$4.79)
(σ, λ)	$(5, 0.2)$	$(2, 0.2)$	$(1.5, 1)$	$(5, 0.6)$	$(1, 0.4)$	$(1, 0.5)$
LSKMBC	76.28	73.17	97.61	80.83	97.07	88.77
(Gauss+Expo)	($\pm$4.78)	($\pm$6.59)	($\pm$1.76)	($\pm$6.35)	($\pm$4.70)	($\pm$4.56)
(σ, λ)	$(2, 0.2)$	$(2, 0.6)$	$(0.2, 0.2)$	$(3, 1)$	$(1, 0.6)$	$(2, 0.4)$
LSKMBC	77.10	72.77	97.81	82.33	97.60	89.16
(Cauchy)	($\pm$5.40)	($\pm$6.55)	($\pm$1.65)	($\pm$6.51)	($\pm$4.21)	($\pm$4.37)
(σ, λ)	$(5, 0.2)$	$(3, 0.4)$	$(2, 0.7)$	$(3, 0.5)$	$(2, 0.3)$	$(3, 0.8)$

and vice-versa. The significance to which the two classifiers are different is obtained from the measure t. As provided in [7] (as obtained from the Student's cumulative t-distribution function with $N-1(=29)$ degrees of freedom), if $|t| > 2.0452$ then the classifiers are different from each other with a confidence level of 97.5%, and the classifiers are different with a 95% confidence level if $|t| > 1.6991$. Table 3 summarizes the t-statistic as obtained by the resampled paired t-test. In the 'Wdbc' dataset, the best variant of LSKMBC (Gaussian kernel) is worse than the best SVM (Gaussian kernel) with a confidence 58.24% ($t = 0.21$), however, the LSKMBC performs significantly better (with a probability very close to unity) than LSSVM. In all other datasets, we observe that the best variant of LSKMBC significantly outperforms the best variant of SVM and LSSVM.

3.2.3 Adult and Web Datasets

In Platt [13], two different datasets namely 'adult' and 'web' datasets were used. In the adult dataset, the task is to classify households whether having annual income greater than USD 50K or not based on the 14 different census fields including eight categorical variables. The data is transformed into 123 sparse binary attributes where six continuous variables are quantized into quantiles. In the web dataset, the task is to predict whether a web page belongs to particular category or not based on the presence of 300 different keywords. Thus the web dataset also has 300 sparse binary attributes. The original 'adult' and 'web' datasets consist of 32562 and 49749 training samples. Since we used Matlab quadratic programming library to directly optimize the objective functional of the LSKMBC, the large number of samples could not be accommodated due to the limitation of the virtual memory. Originally Platt [13] used sequential minimal optimization for the larger datasets,

Table 3: Resampled paired t-test scores in terms of t-statistic comparing the three variants of LSKMBC with the two variants of SVM and LSSVM on the datasets described in Table 1.

Classifier Pair	Pima	Bupa	Wdbc	Wpbc	Iris	Ecoli
LSKMBC (Gauss) SVM (Gauss)	-3.94	-7.33	0.21	-3.33	-3.26	2.37
LSKMBC (Gauss) SVM (Poly)	-7.0	-3.81	-6.93	-9.59	-2.72	-2.19
LSKMBC (Gauss) LSSVM	-4.27	-8.10	-14.73	-4.75	-17.83	-12.10
LSKMBC (GaussExpo) SVM (Gauss)	1.73	-8.00	1.48	-0.41	-5.43	-2.54
LSKMBC (GaussExpo) SVM (Poly)	-2.03	-4.61	-4.91	-6.41	-4.63	-8.53
LSKMBC (GaussExpo) LSSVM	1.52	-8.56	-14.11	-2.29	-17.81	-13.34
LSKMBC (Cauchy) SVM (Gauss)	-4.84	-6.97	1.09	-2.71	-4.82	-3.25
LSKMBC (Cauchy) SVM (Poly)	-7.4	-3.49	-5.38	-7.45	-4.85	-6.56
LSKMBC (Cauchy) LSSVM	-4.9	-7.91	-14.68	-3.90	-17.94	-13.79

however, we have not used any equivalent implementation of the LSKMBC. Platt [13] used nested sets of training samples for both the 'adult' and the 'web' datasets. We used the first two subsets of the 'adult' dataset and the first subset of the 'web' dataset. In Table 4, we show the respective details of the 'adult' and 'web' datasets that we used for the experimentation. These datasets are highly unbalanced (samples from one class largely dominate the dataset) and SVM is able to perform classification with high accuracy even for these unbalanced datasets. We compare the performance of the LSKMBC with that of SVM and LSSVM for different types of kernels. In Table 5 we report the results over the test samples for all classifiers with the best parameter settings. Here also we observe that the LSKMBC performs better than the SVM and LSSVM although the differences in performance scores are not so significant. However this shows that the LSKMBC is able to perform well even for the unbalanced datasets such as adult and web datasets.

Table 4: Description of the nested subsets of the 'adult' and 'web' datasets.

Dataset	number of attributes	number of training samples	number of test samples
Adult1	123	1605	30956
Adult2	123	2265	30296
Web1	300	2477	47272

Table 5: Best classification performance on the 'adult' and 'web' test datasets for SVM, LSSVM and LSKMBC. Corresponding best parametric settings of each classifier is also reported.

Classifier	SVM (Gaussian) (σ^2, C)	SVM (Cubic) (C)	SVM (Linear) (C)	LSSVM (Gaussian) (σ^2, γ)	LSKMBC (Gaussian) (σ, λ)	LSKMBC (Gauss+Expo) (σ, λ)	LSKMBC (Cauchy) (σ, λ)
Adult1	84.22 (10,1)	80.15 (0.01)	84.26 (0.05)	82.1 (20,20)	84.14 (10,0.8)	84.14 (10,0.8)	84.28 (10,0.8)
Adult2	84.33 (10,1)	79.18 (0.01)	84.44 (0.05)	81.42 (20,20)	84.55 (10,0.8)	84.55 (10,0.8)	84.69 (10,0.8)
Web1	97.96 (10,5)	97.72 (0.01)	97.74 (1)	97.98 (100,5)	98.1 (5,0.2)	97.88 (7,0.2)	98.1 (5,0.2)

4. Conclusions

We presented a least square kernel machine classifier with box constraint which employs uniform hyperpriors constrained within a hypercube defined by a given margin. The margin acts as a model selection parameter. We have shown the relationship of the classifier with the existing least square kernel classifiers such as RLSC and LSSVM. We experimentally demonstrated the effectiveness of the classifier and shown that it is able to outperform the SVM and LSSVM on certain real-life datasets. It may be mentioned here that the LSSVM and RLSC were developed to improve the performance in terms of speed. LSKMBC in that sense is slower than the LSSVM because LSKMBC handles the quadratic optimization task like SVM. However, from the classification performance perspective, LSKMBC can outperform both SVM and LSSVM on several datasets. We also mention that the LSKMBC is not necessarily restricted to Mercer kernels. We used long-tailed kernel functions such as Cauchy kernels and observed that the performance of the LSKMBC significantly improves with long-tailed kernel functions.

We have experimented with three different kernels. Since we do not need the Mercer condition to be satisfied in the kernel design, some other kernel functions can also be investigated which constitutes a scope of further study. We also formulated the multi-class classification using one-against-all strategy. However, the objective functional can possibly be designed to incorporate the multi-class classification task as a part of the future study. Sequential minimal optimization (SMO) is used to implement SVM where the sparsity of the support vectors is exploited for efficient implementation for handling larger datasets. The same can be used to effectively implement LSKMBC where larger datasets can be handled efficiently. We designed LSKMBC by constraining the outcome of the classifier which is essentially translated into certain smoothing of the estimated densities. There may not be any straight-forward mechanism for applying this heuristic to other different forms of classifiers. However, one concept is that if constraining the outcome of a classifier translates into the smoothing of the estimated density then possibly this heuristic can be applied.

References

[1] G. A. Babich and O. I. Camps, Weighted Parzen windows for pattern classification, IEEE Trans Pattern Analysis Machine Intelligence, 18, pp. 567-570, 1996.

[2] T. D. Bie, G. R. G. Lanckriet, and N. Cristianini, Convex tuning of the soft margin parameter, UCB-CSD-03-1289, Berkeley, California, USA: University of California, Computer Science Division (EECS), 2003.

[3] C. L. Blake and C. J. Merz, UCI repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.
[4] S. Canu, Y. Grandvalet, V. Guigue, and A. Rakotomamonjy, SVM and kernel methods matlab toolbox, Perception Systmes et Information, INSA de Rouen, Rouen, France, 2005.
[5] P.-H. Chen, C.-J. Lin and B. Schölkopf, A tutorial on nu-support vector machines, Applied Stochastic Models in Business and Industry, 21, pp. 111-136, 2005.
[6] C. Cortes and V. Vapnik, Support vector networks, Machine Learning, 20, pp. 1-25, 1995.
[7] T. G. Dietterich, Approximate statistical test for comparing supervised classification learning algorithms, Neural Computation, 10, pp. 1895-1923, 1998.
[8] D. W. J. Hosmer and S. Lemeshow, Applied logistic regression (2nd ed.), John Wiley, USA, 2000.
[9] D. W. Marquardt, Generalized inverses, ridge regression, biased linear estimation, and nonlinear estimation, Technometrics, 12, pp. 591-612, 1970.
[10] B. A. Olshausen and D. J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature, 381, pp. 607-609, 1996.
[11] M. Osborne, B. Presnell, and B. Turlach, On the LASSO and its dual, Journal Comput. Graphical Statist., 9, pp. 319-337, 2000.
[12] K. Pelckmans, J. A. K. Suykens, T. Van Gestel, J. De Brabanter, L. Lukas, B. Hamers, B. De Moor, and J. Vandewalle, LS-SVMlab toolbox user's guide, 02-145, http://www.esat.kuleuven.ac.be/sista/lssvmlab/, Department of Electrical Engineering, ESAT-SCD-SISTA, Katholieke Universiteit Leuven, Belgium, 2003.
[13] J. C. Platt, Sequential minimal optimization : A fast algorithm for training support vector machines MSR-TR-98-14, Microsoft Research, USA, 1998.
[14] R. Rifkin and A. Klautau, In defense of one-vs-all classification, Journal of Machine Learning Research, 5, pp. 101-141, 2004.
[15] R. Rifkin, G. Yeo and T. Poggio, Regularized least square classification, In J. Suykens, G. Horvath, S. Basu, C. Micchelli, and J. Vandewalle, Advances in learning theory: Methods, model and applications, NATO science series III: Computer and system sciences, vol. 190, pp. 131-153, IOS Press, Amsterdam, 2003.
[16] V. Roth, The generalized LASSO, IEEE Trans. Neural Networks, 15(1), pp. 16-28, 2004.
[17] C. Saunders, A. Gammerman, and V. Vovk, Ridge regression learning algorithm in dual variables, In Proceedings of the 15th international conference on machine learning (icml98), 1998.
[18] J. Shawe-Taylor and N. Cristianini, An introduction to support vector machines and other kernel-based learning methods, Cambridge University Press, UK, 2000.
[19] A. J. Smola and B. Schölkopf, A tutorial on support vector regression NC-TR-98-030, Royal Holloway College, University of London, NeuroCOLT, UK, 1998.
[20] J. A. K. Suykens and J. Vandewalle, Least square support vector machine classifiers, Neural Processing Letters, 9, pp. 293-300, 1999.
[21] R. Tibshirani, Regression shrinkage and selection via LASSO, Journal Royal Statistical Society, Series B, 58, pp. 267-288, 1996.
[22] A. N. Tikhonov and V. Y. Arsenin, Solutions of ill-posed problems, W. H. Winston, Washington, D. C., 1977.
[23] M. Tipping, Sparse Bayesian learning and the relevance vector machine, Journal Machine Learning Research, 1, pp. 211-244, 2001.
[24] T. Van Gestel, J. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J. Vandewalle, Benchmarking least squares support vector machine classifiers, Machine Learning, 54, pp. 5-32, 2004.
[25] V. Vapnik, Statistical learning theory, Springer-Verlag, New York, USA, 1998.