Journal of the Korean Data & Information Science Society
2016, **27**(2), 549–557

http://dx.doi.org/10.7465/jkdi.2016.27.2.549
한국데이터정보과학회지

LS-SVM for large data sets[†]

Hongrak Park[1] · Hyungtae Hwang[2] · Byungju Kim[3]

[1]Spring Information Technology
[2]Department of Applied Statistics, Dankook University
[3]Department of Computer Engineering, Youngsan University

Received 11 February 2016, revised 9 March 2016, accepted 21 March 2016

Abstract

In this paper we propose multiclassification method for large data sets by ensembling least squares support vector machines (LS-SVM) with principal components instead of raw input vector. We use the revised one-vs-all method for multiclassification, which is one of voting scheme based on combining several binary classifications. The revised one-vs-all method is performed by using the hat matrix of LS-SVM ensemble, which is obtained by ensembling LS-SVMs trained using each random sample from the whole large training data. The leave-one-out cross validation (CV) function is used for the optimal values of hyper-parameters which affect the performance of multiclass LS-SVM ensemble. We present the generalized cross validation function to reduce computational burden of leave-one-out CV functions. Experimental results from real data sets are then obtained to illustrate the performance of the proposed multiclass LS-SVM ensemble.

Keywords: Ensemble, generalized cross validation function, least squares support vector machine, multiclassification, one-vs-all method, principal components, random sample.

1. Introduction

The support vector machine (SVM), firstly developed by Vapnik (1995, 1998) and his group at AT&T Bell Laboratories, has been successfully applied to a number of real world problems related to the classification and regression. Despite of lots of successful applications of SVM in regression and classification problems, training SVM requires to solve a quadratic programming problem, which is computationally formidable for the large data. Least squares SVM (LS-SVM) is least squares version of SVM and was initially introduced by Suykens and Vanderwalle (1999a). LS-SVM has been proved to be a very appealing and promising method (Suykens *et al.*, 2001; Seok, 2010; Shim and Seok, 2014; Hwang, 2015; Shim and Hwang, 2015).

Multiclassification is typically performed by using the voting scheme based on combining several binary classifications (Hastie and Tibshirani, 1998; Ghosh, 2002), which includes

[†] This research was supported by Software Convergence Cluster Program, through the Ministry of Science, ICT and Future Planning (2015-GSWC-D4).
[1] CEO, Spring Information Technology, Gyeongsan 38685, Korea.
[2] Professor, Department of Statistics, Dankook University, Yongin 16890, Korea.
[3] Corresponding author: Professor, Department of Computer Engineering, Youngsan University, Yangsan 50510, Korea. E-mail: bjkim@ysu.ac.kr

one-versus-all method and one-versus-one (pairwise) method. For SVM Weston and Watkins (1998) proposed the multiclassification without using the combination of binary classifications. For LS-SVM Suykens and Vanderwalle (1999b) proposed multiclassification in a step with linear system composed of linear equations from each binary classifications.

To use LS-SVM for large data, Espinoza *et al.* (2005) proposed the fixed size LS-SVM with sparse approximation of nonlinear feature mapping functions which are induced by kernel functions computed based on Nyström approximations (Williams and Seeger, 2001) and quadratic Renyi entropy (Girolami, 2003). Hwang (2015) combined LS-SVMs on random subsamples of large training data set for the multiclassification of test data set.

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of variables into a set of values of linearly uncorrelated principal components. The number of principal components is less than or equal to the number of variables of raw data. The first principal component has the largest variance and each succeeding component in turn has the highest variance possible under the constraint of orthogonality to the preceding components. It is known that PCA has lots of applications including contraction of data, denoising and regression under multicollinearity. For brief reviews of PCA see Jolliffe (2002).

In this paper we propose LS-SVM for large data sets, which is performed by ensenmbling LS-SVMs which are trained using each disjoint random sample from the whole large training data. In the training data set and the test data set, input vector consist of small number of principal components obtained by applying eigen vectors of original raw input vector. We present the leave-one-out cross validation (CV) function for the optimal values of hyperparameters which affect the performance of multiclassification and we obtain the generalized cross validation (GCV) function for the approximate of CV function.

The remainder of paper is organized as follows. In Section 2 we propose a method of ensembling LS-SVMs. In Section 3 we propose the multiclassification method by using principal components and ensembling LS-SVMs. In Section 4 we illustrate the performance of the proposed method through four real data sets. Section 5 contains the conclusions.

2. Ensembling LS-SVM for large data

2.1. LS-SVM for regression

For the training data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, with each input vector $\boldsymbol{x}_i \in R^d$, the response $y_i \in R$, and the test data by $\boldsymbol{x}_t$, we consider the nonlinear regression function given as the form of $f(\boldsymbol{x}) = \boldsymbol{w}'\phi(\boldsymbol{x}) + b$, where b is a bias. Here $\phi : R^d \to R^{d_f}$ is the nonlinear feature mapping function which maps the input space to the higher dimensional feature space, where the dimension d_f defined in an implicit way.The optimization problem of LS-SVM is defined as follows:

$$\min \frac{1}{2}\boldsymbol{w}'\boldsymbol{w} + \frac{C}{2}\sum_{i=1}^{n} e_i^2 \tag{2.1}$$

subject to $e_i = y_i - \boldsymbol{w}'\phi(\boldsymbol{x}_i) - b,\ i = 1, \cdots, n,$
where $C > 0$ is a penalty parameter which controls the tradeoff between the goodness-of-fit on the data and $\boldsymbol{w}'\boldsymbol{w}$.

From (2.1) the Lagrangian function is constructed as follws:

$$L = \frac{1}{2}\boldsymbol{w}'\boldsymbol{w} \ + \ \frac{C}{2}\sum_{i=1}^{n} e_i^2 - \sum_{i=1}^{n} \alpha_i(e_i - y_i + \boldsymbol{w}'\phi(\boldsymbol{x}_i) + b), \tag{2.2}$$

where α_i's are the Lagrangian multipliers.

From the conditions for optimality we have,

$$\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{w}} &= 0 \ \rightarrow \ \boldsymbol{w} - \sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) = \boldsymbol{0} \\
\frac{\partial L}{\partial b} &= 0 \ \rightarrow \ \sum_{i=1}^{n} \alpha_i = 0 \\
\frac{\partial L}{\partial e_i} &= 0 \ \rightarrow \ e_i - \alpha_i/C = 0, \ i = 1, \cdots, n \\
\frac{\partial L}{\partial e_i} &= 0 \ \rightarrow \ \boldsymbol{w}'\phi(\boldsymbol{x}_i) + b + e_i - y_i = 0, \ i = 1, \cdots, n,
\end{aligned}$$

which are equivalent to the linear equations as follows:

$$\sum_{j=1}^{n} K(\boldsymbol{x}_i, \boldsymbol{x}_j)\alpha_i + \alpha_i/C + b = y_i, i = 1, \cdots, n, \text{ and } \sum_{i=1}^{n} \alpha_i = 0, \tag{2.3}$$

where $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)'\phi(\boldsymbol{x}_j)$, which is obtained by the application of Mercer's conditions (1909).

From the linear equation (2.3) the bias estimate and optimal values of Lagrangian multipliers, $\widehat{b}$ and $\widehat{\alpha}_i$'s can be obtained. The predicted regression function given $\boldsymbol{x}_t \in R^d$ is obtained as

$$\widehat{y}(\boldsymbol{x}_t) = K(\boldsymbol{x}_t, \boldsymbol{x})\widehat{\boldsymbol{\alpha}} + \widehat{b} = H_t\boldsymbol{y}, \tag{2.4}$$

where $H_t = (K(\boldsymbol{x}_t, \boldsymbol{x}), \boldsymbol{1})H_0$, $\boldsymbol{x} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)' \in R^{n\times d}$, $\boldsymbol{y} = (y_1, \cdots, y_n)' \in R^n$,
$K = K(\boldsymbol{x}, \boldsymbol{x})$ and $H_0 = \begin{pmatrix} (K + I/C)^{-1} - (K+I/C)^{-1}\boldsymbol{1}(\boldsymbol{1}'(K+I/C)^{-1}\boldsymbol{1})\boldsymbol{1}'(K+I/C)^{-1} \\ (\boldsymbol{1}'(K+I/C)^{-1}\boldsymbol{1})^{-1}\boldsymbol{1}'(K+I/C)^{-1} \end{pmatrix}$.

It is known that it can be easily shown that Lagrangian multipliers of LS-SVM for binary classification are identical to product of diagonal matrix of $\boldsymbol{y}$ and Lagrangian multipliers of LS-SVM for regression obtained from equation (2.3), when $\boldsymbol{y}$ consists of class labels -1 and 1. That is, if y_i =-1 or 1, then $\widehat{y}(\boldsymbol{x}_{ti})$'s obtained by LS-SVM for regression and LS-SVM for binary classification are identical. Thus, for the binary classification, each observation of the test data $\boldsymbol{x}_t$ can be classified into either class according to the sign of $\widehat{y}(\boldsymbol{x}_t)$ in (2.4).

Instead of LS-SVM for binary classification, we use LS-SVM for regression to approximate the leave-one-out cross validation function easily.

The performance of LS-SVM is affected by hyper-parameters, the penalty parameter $C > 0$ and the kernel parameters. To select the optimal values of hyper-parameters of multiclass

LS-SVM, we use the leave-one-out cross validation (LOO-CV) function as follows:

$$CV(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i^{(-i)}(\boldsymbol{\theta}))^2, \tag{2.5}$$

where $\boldsymbol{\theta}$ is a candidate set of hyper-parameters and $\widehat{y}_i^{(-i)}(\boldsymbol{\theta})$ is the predicted value of y_i obtained from data without the ith observation. Since for each candidate set of hyper-parameters, $\widehat{y}_i^{(-i)}(\boldsymbol{\theta})$ for $i = 1, \cdots, n$, should be evaluated, selecting parameters using LOO-CV function is computationally formidable. By using leaving-out-one lemma (Wahba, 1990) and the first order of Taylor expansion, the ordinary cross validation function is obtained as follows:

$$OCV(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1-\widehat{y}_i(\boldsymbol{\theta})}{1-\dfrac{\partial\widehat{y}_i}{\partial y_i}}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1-\widehat{y}_i(\boldsymbol{\theta})}{1-h_{ii}(\boldsymbol{\theta})}\right)^2, \tag{2.6}$$

where $h_{ii}(\boldsymbol{\theta})$ for $i = 1, \cdots, n$, is the ith diagonal element of the hat matrix $H = H(\boldsymbol{x}, \boldsymbol{x})$ such that $\widehat{\boldsymbol{y}} = H\boldsymbol{y}$. By averaging the residuals in (2.6) by$(1 - trace(H)/n)$, the generalized cross validation (GCV) function is obtained as follows:

$$GCV(\boldsymbol{\theta}) = \frac{n\sum_{i=1}^{n}(1-\widehat{y}_i(\boldsymbol{\theta}))^2}{(n - trace(H))^2}. \tag{2.7}$$

2.2. Ensembling LS-SVM with principal components

The training data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is given, with each input vector $\boldsymbol{x}_i \in R^{d_c}$ which consists of d_c principal components, the output $y_i \in R$, and the test data by $\boldsymbol{x}_t$. Here $\boldsymbol{x}_i$ and $\boldsymbol{x}_t$ consist of d_c principal components computed from PCA of the original input vector of training data set.

In LS-SVM, we need the inverse of $(K(\boldsymbol{x}, \boldsymbol{x}) + I/C)^{-1}$, which is almost impossible for very large data set of size N. Here we propose a method of ensembling LS-SVM for large data. Instead training one LS-SVM on the whole data at once, we train M LS-SVMs using each random sample of size n from the whole training data and aggregate them to obtain the predicted regression function for the test data $\boldsymbol{x}_t \in R^{d_c}$ as follows:

$$\widehat{y}(\boldsymbol{x}_t) = \frac{1}{M}\sum_{j=1}^{M}(K(\boldsymbol{x}_t, \boldsymbol{x}^j)\widehat{\boldsymbol{\alpha}}^j + \widehat{b}^j), \tag{2.8}$$

where M is a number of random samples, $\boldsymbol{\alpha}^j$ and b^j are computed from the linear equation (2.3) using the jth disjoint random sample $(\boldsymbol{x}^j, \boldsymbol{y}^j)$ from the whole training data such that $(\boldsymbol{x}, \boldsymbol{y}) = \bigcup_{j=1}^{M}(\boldsymbol{x}^j, \boldsymbol{y}^j)$ and $(\boldsymbol{x}^j, \boldsymbol{y}^j)\bigcap(\boldsymbol{x}^k, \boldsymbol{y}^j) = \{\}$ for $j \neq k$. Ensembling LS-SVM is inspired by the basic idea of the bagging (Breiman, 1996), known to improve the stability and reduces the variance and help to avoid overfitting.

In ensembling LS-SVM, we obtain the inverse of $\left(K(\boldsymbol{x}^j, \boldsymbol{x}^j) + I/C\right)^{-1}$ for $n << N$, which enables to train LS-SVM using large data.

3. Multiclassification

3.1. Multiclass LS-SVM

In this section we give simple overview on multiclassification by LS-SVM using one-against-all method. The training data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is given, with each input vector $\boldsymbol{x}_i \in R^d$ and the class label $y_i \in \{1, 2, \cdots, m\}$, where m is number of classes. For multiclassification using one-against-all method, we first transform $\boldsymbol{y}$ into $n \times m$ matrix $\boldsymbol{Y}$ which consists of 1 and -1, where $Y_{ik} = 1$ and $Y_{il} = -1$ for $k \neq l$ implies ith observation belongs to the kth class. Then we have m LS-SVMs for binary classification with $\{(\boldsymbol{x}_i, Y_{ik})\}_{i=1}^n$ for $k = 1, \cdots, m$.

From the linear equation such that

$$\begin{bmatrix} K + \boldsymbol{I}/C & \mathbf{1} \\ \mathbf{1}' & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\cdot k} \\ b_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{Y}_{\cdot k} \\ 0 \end{bmatrix}, \ k = 1, 2, \cdots, m, \tag{3.1}$$

where $\boldsymbol{Y}_{\cdot k}$ is the kth column of $\boldsymbol{Y}$, the estimate of bias and the optimal Lagrangian multipliers, $\widehat{b}_k$ and $\widehat{\boldsymbol{\alpha}}_{\cdot k}$ can be obtained. For the test data $\boldsymbol{x}_t$ we have,

$$\widehat{Y}_k(\boldsymbol{x}_t) = K(\boldsymbol{x}_t, \boldsymbol{x})\widehat{\boldsymbol{\alpha}}_{\cdot k} + \widehat{b}_k, \ k = 1, \cdots, m. \tag{3.2}$$

Then the test data $\boldsymbol{x}_t$ is classified into the kth class for $k = 1, 2, \cdots, m$, if $sign(\widehat{Y}_k(\boldsymbol{x}_t)) = 1$ and $sign(\widehat{Y}_l(\boldsymbol{x}_t)) = -1$ for $k \neq l$.

3.2. Model selection of multiclass LS-SVM

The performance of multiclass LS-SVM is affected by hyper-parameters, the penalty parameter C and the kernel parameters. To select the optimal values of hyper-parameters of multiclass LS-SVM, we use the leave-one-out cross validation (LOO-CV) function as follows:

$$CV(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}(Y_{ik_i} - \widehat{Y}_{ik_i}^{(-i)}(\boldsymbol{\theta}))^2,$$

where $\boldsymbol{\theta}$ is a candidate set of hyper-parameters and $\widehat{Y}_{im_i}^{(-i)}(\boldsymbol{\theta})$ is the predicted value of Y_{ik_i} obtained from data without the ith observation. Here k_i is the column number of the ith row of $\boldsymbol{Y}$ such that $Y_{ik_i} = 1$, which implies that the ith observation is classified into the k_ith class. Since for each candidate set of hyper-parameters, $\widehat{Y}_{ik_i}^{(-i)}(\boldsymbol{\theta})$ for $i = 1, \cdots, n$, should be calculated, selecting the optimal values of hyper-parameters using LOO-CV function is computationally formidable. The ordinary cross validation (OCV) function is obtained as follows:

$$OCV(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1 - \widehat{Y}_{ik_i}(\boldsymbol{\theta})}{1 - \dfrac{\partial \widehat{Y}_{ik_i}}{\partial Y_{im_i}}}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1 - \widehat{Y}_{ik_i}(\boldsymbol{\theta})}{1 - h_{ii}(\boldsymbol{\theta})}\right)^2, \tag{3.3}$$

where $h_{ii}(\boldsymbol{\theta})$ for $i = 1, \cdots, n$, is the ith diagonal element of the hat matrix $H = H(\boldsymbol{x}, \boldsymbol{x})$. By averaging the residuals in (3.4) by $(1 - trace(H)/n)$, the generalized cross validation (GCV)

function is then obtained as follows:

$$GCV(\boldsymbol{\theta}) = \frac{n\sum_{i=1}^{n}(1-\widehat{Y}_{ik_i}(\boldsymbol{\theta}))^2}{(n-trace(H))^2}. \tag{3.4}$$

3.3. Ensembling multiclass LS-SVM

For the multiclassification of test data $\boldsymbol{x}_t$ we use a hat matrix to avoid solving m (number of classes) linear equations in (3.1). Here (3.2) can be reexpressed as

$$\begin{aligned}\widehat{Y}_k(\boldsymbol{x}_t) &= K(\boldsymbol{x}_t,\boldsymbol{x})\widehat{\boldsymbol{\alpha}}_{,k}+\widehat{b}_k = (K(\boldsymbol{x}_t,\boldsymbol{x}),1)\begin{pmatrix}\widehat{\boldsymbol{\alpha}}_{.k}\\ \widehat{b}_k\end{pmatrix}\\ &= (K(\boldsymbol{x}_t,\boldsymbol{x}),1)\begin{pmatrix}(K+I/C)^{-1}-(K+I/C)^{-1}\mathbf{1}(\mathbf{1}'(K+I/C)^{-1}\mathbf{1})\mathbf{1}'(K+I/C)^{-1}\\ (\mathbf{1}'(K+I/C)^{-1}\mathbf{1})^{-1}\mathbf{1}'(K+I/C)^{-1}\end{pmatrix}\boldsymbol{Y}_{.k}\\ &= (K(\boldsymbol{x}_t,\boldsymbol{x}),1)H_0(\boldsymbol{x})\boldsymbol{Y}_{.k} = H(\boldsymbol{x}_t,\boldsymbol{x})\boldsymbol{Y}_{.k} \text{ for } k=1,\cdots,m.\end{aligned} \tag{3.5}$$

Since $H(\boldsymbol{x}_t,\boldsymbol{x})$ does not depend on $\boldsymbol{Y}_{.k}$ we can express $\widehat{Y}(\boldsymbol{x}_t)$ for the test data $\boldsymbol{x}_t$ as follows:

$$\widehat{Y}(\boldsymbol{x}_t) = H(\boldsymbol{x}_t,\boldsymbol{x})\boldsymbol{Y}, \tag{3.6}$$

where $\widehat{Y}(\boldsymbol{x}_t) = (\widehat{Y}_1(\boldsymbol{x}_t),\cdots,\widehat{Y}_m(\boldsymbol{x}_t))$.

Thus, we do not need to solve linear equations m times using (3.1) but once given as (3.6).

For large data we divide the whole training data into M random samples such that the whole training data $(\boldsymbol{x},\boldsymbol{Y}) = \bigcup_{j=1}^{M}(\boldsymbol{x}^j,\boldsymbol{Y}^j)$ and $(\boldsymbol{x}^j,\boldsymbol{Y}^j)\bigcap(\boldsymbol{x}^k,\boldsymbol{Y}^k) = \{\}$ for $j\neq k$. Then $\widehat{Y}(\boldsymbol{x}_t)$ for the test data $\boldsymbol{x}_t$ can be written as follows:

$$\widehat{Y}(\boldsymbol{x}_t) = \frac{1}{M}\sum_{j=1}^{M}(K(\boldsymbol{x}_t,\boldsymbol{x}^j),1)H_0(\boldsymbol{x}^j)\boldsymbol{Y}^j. \tag{3.7}$$

The optimal values of hyper-parameters for training the individual LS-SVM using $(\boldsymbol{x}^j,\boldsymbol{Y}^j)$ for $j=1,\cdots,M$, are selected by GCV function (3.4).

4. Numerical Studies

Through 4 real data sets available from UCI Machine Learning Depository (http://archive.ics.uci.edu/ml) and MNIST handwritten digit database (http://yann.lecun.com/exdb/mnist) - Wine data set, Glass data set, Sensor readings4 data set and Handwritten digit data set - we illustrate the performance of multiclass LS-SVM ensemble (LS-SVME). As input vector we use principal components instead of raw input vector in the original data set. The radial basis function kernel is utilized for LS-SVM in numerical studies.

To illustrate the performance of multiclass LS-SVME, we run multiclass LS-SVM and the classification and regression trees (CART, Breiman *et al.*, 1984), the bootstrap aggregation (Breiman, 1996) of 50 CARTs, and compare misclassification rates each other. We randomly divide the whole data set into the training data set and the test data set. The averages of 50

misclassification rates from multiclass LS-SVM, CART, and LS-SVME are obtained from each test data set. The penalty parameter and bandwidth parameter, C and σ^2 are obtained from training data set by GCV function (3.4). We use 10-fold cross validation method for the model selection of CART.

Wine data set of 3 classes obtained from results of wines grown in the same region in Italy but derived from three different cultivars, consists of 12 input variables and 178 observations. There are 144 observations in the training data set and 34 observations in the test data set. We use 8 principal components and 2 random samples of the whole training data set for multiclass LS-SVME. The averages and standard errors of 50 misclassification rates on 50 test data sets are shown in Table 4.1. From the results we can see that multiclass LS-SVME have the best multiclassification performance on this data set.

Glass data set of 6 classes from the study of classification of types of glass motivated by criminological investigation, consists of 9 input variables and 214 observations. There are 140 observations in the training data set and 74 observations in the test data set. We use 6 principal components and 2 random samples of the whole training data set for multiclass LS-SVM ensemble. The averages and standard errors of 50 misclassification rates on 50 test data sets are shown in Table 4.1. From the results we can see that multiclass LS-SVM have the better multiclassification performance than that of CART on this data set.

Due to out of memory of MATLAB R2006b which is implemented for the numerical studies we cannot train multiclass LS-SVM using the whole training data sets of following two examples. Instead we use CART.

Sensor readings4 data set of 4 classes from results of robot navigates through the room following the wall in the clockwise direction for 4 rounds using 24 ultrasound sensors arranged circularly around its waist, consists of 4 input variables and 5456 observations. There are 5000 observations in the training data set and 456 observations in the test data set. We use 4 principal components and 5 random samples of the whole training data set for multiclass LS-SVME. The averages and standard errors of 50 misclassification rates on 50 test data sets are shown in Table 4.1. From the results we can see that multiclass LS-SVME have the better multiclassification performance than that of CART on this data set.

Handwritten digit data set of 10 classes consists of digit $0, 1, \cdots, 9$ and 6000 observations of 28×28 pixel image as shown in Figure 4.1. There are 5000 observations in the training data set and 1000 observations in the test data set. We use 10 principal components and 10 random samples of the whole training data set for multiclass LS-SVME. The averages and standard errors of 100 misclassification rates on 100 test data sets are shown in Table 4.1. From the results we can see that multiclass LS-SVME have the best multiclassification performance on this data set.

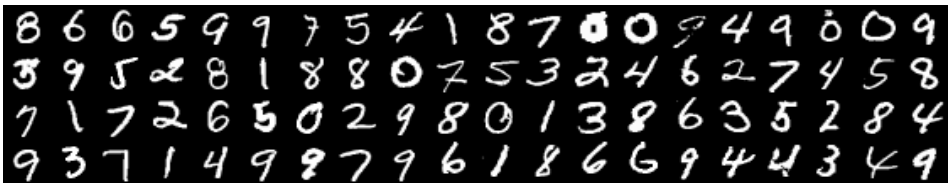

Figure 4.1 Some image data in handwritten digit data set

Table 4.1 The averages of misclassification error rates (standard error in parenthesis)

	LS-SVM	LS-SVM ensemble	CART	Bagging CART
Wine	0.0488	0.0482	0.0929	0.0576
N=144,	(0.0055)	(0.0048)	(0.0049)	(0.0062)
Nt=33		M=2		
Glass	0.3033	0.3515	0.4678	0.3341
N=140	(0.0079)	(0.085)	(0.0099)	(0.0066)
Nt=74		M=2		
Sensor readings4		0.0399	0.0518	0.0267
N=5000		(0.0012)	(0.0014)	(0.001)
Nt=456		M=5		
Handwritten digit		0.0642	0.3615	0.0884
N=5000		-0.0011	(0.0022)	(0.0011)
Nt=1000		M=10		

5. Conclusions

Through the examples we showed that ensembling multiclass LS-SVM with principal components shows the good results, which is simple modeling of the multiclassification problem for large data sets. Especially the proposed method showed good multiclassification performance on the image data with many input variables. In future work, we study the optimal numbers of random samples and principal components for multiclass LS-SVM ensemble.

References

Breiman, L. (1996). Bagging predictors. *Machine Learning*, **24**, 123-140.

Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). *Classification and regression trees*, Wadsworth, New York.

Espinoza, M., Suykens, J. A. K. and De Moor, B. (2005). Load forecasting using least squares Support vector machines. *Lecture Notes in Computer Science*, **3512**, 1018-1026.

Girolami, M. (2003). Orthogonal series density estimation and kernel eigenvalue problem. *Neural Computation*, **14**, 669-688.

Ghosh, J. (2002). Multiclassifier systems: Back to the future. *Lecture Note in Computer Science*, **2364**, 1-15.

Hastie, T. and Tibshirani, R. (1998). Classification by pairwise coupling. *The Annals of Statistics*, **26**, 451-471.

Hwang, H. (2015). Multiclass LS-SVM ensemble for large data. *Journal of the Korean Data & Information Science Society*, **26**, 1557-1563.

Jolliffe, I. T. (2002). *Principal component analysis (2nd edition)*, Springer, New York.

Mercer, J. (1909). Functions of positive and negative type and their connection with theory of integral equations. *Philosophical Transactions of Royal Society A*, **209**, 415-446.

Seok, K. H. (2010). Semi-supervised classification with LS-SVM formulation. *Journal of the Korean Data & Information Science Society*, **21**, 461-470.

Shim, J. and Hwang, C. (2015). Varying coefficient modeling via least squares support vector regression. *Neurocomputing*, **161**, 254-259.

Shim, J. and Seok, K. H. (2014). A transductive least squares support vector machine with the difference convex algorithm. *Journal of the Korean Data & Information Science Society*, **25**, 455-464.

Suykens, J. A. K. and Vanderwalle, J. (1999a). Least square support vector machine classifier, *Neural Processing Letters*, **9**, 293-300.

Suykens, J. A. K. and Vandewalle, J. (1999b). Multiclass least squares support vector machines. In *Proceeding of the International Joint Conference on Neural Networks*, 900-903, IEEE, Washington, D. C..

Suykens, J. A. K., Vandewalle, J. and DeMoor, B. (2001). Optimal control by least squares support vector machines. *Neural Networks*, **14**, 23-35.

Vapnik, V. N. (1995). *The nature of statistical learning theory*, Springer, New York.

Vapnik, V. N. (1998). *Statistical learning theory*, Springer, New York.

Wahba, G. (1990). *Spline models for observational data, CMMS-NSF Regional Conference Series in Applied Mathematics*, **59**, SIAM, Philadelphia.

Weston, J. and Watkins, C. (1998). *Multi-class SVM, Technical Report 98-04, Royal Holloway University*, London.

Williams, C. K. I. and Seeger, M. (2001). Using the Nyström method to speed up kernel machines. In *Proceeding of Neural Information Processing Systems Conference 13*, 682-699, MIT Press, Cambridge, London.