공학박사학위논문

Intelligent Data Selection and Semi-Supervised Learning for Support Vector Regression

Support Vector Regression 을 위한

지능적 데이터 선택 및 **Semi-Supervised Learning**

2013 년 2 월

서울대학교 대학원

산업공학과 데이터마이닝 전공

김 동 일

Intelligent Data Selection and Semi-Supervised Learning for Support Vector Regression

Support Vector Regression 을 위한
지능적 데이터 선택 및 **Semi-Supervised Learning**

지도교수 조 성 준

이 논문을 공학박사 학위논문으로 제출함

2012 년 12 월

서울대학교 대학원

산업공학과 데이터마이닝 전공

김 동 일

김동일의 박사학위논문을 인준함

2012 년 12 월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

ABSTRACT

Intelligent Data Selection and Semi-Supervised Learning for Support Vector Regression

Dongil Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

Support Vector Regression (SVR), a regression version of Support Vector Machines (SVM), employing Structural Risk Minimization (SRM) principle has become one of the most spotlighted algorithms with the capability of solving nonlinear problems using the kernel trick. Despite of the great generalization performance, there still exist open problems for SVR to overcome. In this dissertation, two major open problems of SVR are studied: (1) training complexity and (2) Semi–Supervised SVR (SS–SVR).

Since the training complexity of SVR is highly related to the number of training data n: $O(n^3)$, training time complexity and $O(n^2)$, the training memory complexity, it makes SVR difficult to be applied to big–sized real–world datasets. In this dissertation, a data selection method, Margin based Data Selection (MDS), was proposed in order to reduce the training

complexity. In order to overcome the training complexity problem, reducing the number of training data is an effective approach. Data selection approach is designed to select important or informative data among all training data. For SVR, the most important data are support vectors. By ε–loss foundation and the maximum margin learning, all support vectors of SVR are located on or outside the ε–tube. With multiple sample learning, MDS estimated the margin for all training data, efficiently. MDS selected a subset of data by comparing the margin and ε. Through the experiments conducted on 20 datasets, the performance of MDS was better than the benchmark methods. The training time of SVR including running time of MDS was with 38% $\sim$ 67% of training time of original datasets. At the same time, the accuracy loss was 0% $\sim$ 1% of original SVR model.

Recently, the size of dataset is getting larger, and data are collected from various applications. Since collecting the labeled data is expensive and time consuming, the fraction of the unlabeled data over the labeled data is getting increased. The conventional supervised learning method uses only labeled data to train. Recently, Semi–Supervised Learning (SSL) has been proposed in order to improve the conventional supervised learning by training the unlabeled data along with the labeled data. In this dissertation, a data generation and selection method for SS–SVR training is proposed. In order to estimate the label distribution of the unlabeled data, Probabilistic Local Reconstruction method (PLR) was employed. In order to get robustness to noisy data, two PLRs (PLR_{local} and PLR_{global}) were employed and the final label distribution was obtained by the conjugation of 2–PLR. Then, training data were generated from the unlabeled data with their the estimated label distribution. The data generation rate was differed by uncertainty of the labeling. After that, MDS was employed to reduce the training complexity increased by the generated data. Through the experiments conducted on 18 datasets, the proposed method could improve about 10% of the accuracy than the conventional supervised SVR, and the training time of the proposed method including the construction of final SVR was less than 25% of benchmark methods.

Two applications are analyzed. For response modeling, SVR based two–stage response modeling, identifying respondents at the first stage and then ranking them according to expected profit at the second stage, was proposed. And MDS was employed in order to reduce the training complexity of two–stage response modeling. The experimental results showed that SVR employed two–stage response model could increase the profit

than the conventional response model. MDS reduced the training complexity of SVR to about 60% of original SVR with minimum profit loss. For Virtual Metrology (VM), the proposed SS–SVR method was applied to a real–world VM dataset by using the unlabeled data with the labeled data for training. Data were collected from two pieces of equipment of the photo process. The experimental results showed the proposed SS–SVR method could improve the accuracy about 8% on average than that of the conventional VM model. The accuracy of proposed method was better than benchmark method while the training time of the proposed method was relatively small than benchmark methods.

.....

Keywords: Data Mining, Machine Learning, Pattern Recognition, Support Vector Machines (SVM), Support Vector Regression (SVR), Semi-Supervised Learning (SSL), Semi-Supervised Support Vector Regression (SS-SVR), Data Selection, Data Generation, Regression, Customer Relationship Management (CRM), Response Modeling, Semiconductor Manufacturing, Virtual Metrology.

Student Number: 2005-20821

Notation

Notations used in this dissertation.

$\mathcal{X}$	: a set of input variables, $\mathcal{X} = \{\mathbf{x}_i \| i = 1, ..., n\}, \quad \mathbf{x}_i \in \mathbb{R}^d$.
$\mathcal{Y}$	: a set of labels, $\mathcal{Y} = \{y_i \| i = 1, ..., n\}, \quad y_i = f(\mathbf{x}_i)$.
$\mathbf{X}$	: $n \times d$ matrix of input data, $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]$
$\mathbf{x}$	: a vector in d-dimensional space, $\mathbf{x} \in \mathbb{R}^d$.
x_i	: i-th element of $\mathbf{x}$.
y	: a label corresponding to input variables.
n	: the number of training data.
f	: a function that links between $\mathbf{x}$ and y, $y = f(\mathbf{x})$.
C	: a cost term of SVR.
$\mathbf{w}$	: a weight vector of SVR.
b	: a bias term of SVR.
ε	: the size of ε–insensitive tube of SVR.
$\xi_i^{(*)}$	: a slack variable of the soft margin problem of SVR.
$\alpha_i^{(*)}$	: a lagrangian multiplier of the dual optimization problem of SVR.
$\mathcal{F}$	: a high dimensional feature space.
Φ	: a mapping to features space, $\Phi : \mathcal{X} \rightarrow \mathcal{F}$.
$K(\cdot, \cdot)$	: a kernel function of SVR, $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$.
$\|\|\mathbf{a} - \mathbf{b}\|\|$	: Euclidean distance between $\mathbf{a}$ and $\mathbf{b}$.
$exp(\cdot)$	: an exponential of components.

q	: the size of working set.
k	: the number of nearest neighbors of k–NN.
$\mathbf{x}_{NN(\mathbf{x}_i,j)}$	: the jth nearest neighbor of $\mathbf{x}_i$.
$y_{NN(\mathbf{x}_i,j)}$	: a label of jth nearest neighbor of $\mathbf{x}_i$.
$\bar{y}_{NN(\mathbf{x}_i)}$	: an average of $y_{NN(\mathbf{x}_i,j)}$ over j.
$S(x,y)$	: a similarity of x and y for HSVM.
s_i	: sparsity of $\mathbf{x}_i$ for the k–NN based method.
v_i	: variability of $\mathbf{x}_i$ for the k–NN based method.
u_i	: uniqueness of $\mathbf{x}_i$ for the k–NN based method.
$\mathbf{x}_*$	: a test data point.
D	: the original dataset.
D_j	: the jth sample set from D.
f_j	: trained SVR from the jth sample set.
R_{ij}	: regression error of a data point $\mathbf{x}_i$ evaluated by f_j.
M_{ij}	: binary marking matrix for ε–DS and VDS.
L_i	: likelihood score of a data point $\mathbf{x}_i$ for VDS.
S	: the expected number of support vectors for VDS.
l	: the number of sample sets.
m	: the number of data in a sample set (m% of the original data).
α	: the accuracy–training time control parameter for MDS.
L	: the labeled data.
L_x	: the input variables of the labeled data.
L_y	: the label of the labeled data.
U	: the unlabeled data.
U_x	: the input variables of the unlabeled data.
$\hat{y}$	: the estimated label of the unlabeled data.
D_I	: the integrated data set.
U_G	: the generated data.
D_S	: the selected data set.
p_i	: data generation probability.
k_{local}	: the number of nearest neighbors for PLR$_{local}$.
k_{global}	: the number of nearest neighbors for PLR$_{global}$.
t	: the number of trials for data generation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data mining discovers hidden information or important patterns from a large size of database. Data mining refers to "the process of exploration and analysis, by automatic or semi–automatic means, of large quantities of data in order to discover meaningful patterns and rules (Berry and Linoff, 1997, 2000; Shmueli et al., 2007)." Methods utilized for data mining are originated from various areas: computer science, statistics, artificial intelligence and machine learning. In the process of data mining, mathematical algorithms are programmed to computers in order to solve the problems, automatically. The main goal of data mining is to answer the real–world problems, such as "who will buy?" for a marketing problem or "which one is fault?" for a manufacturing problem. As the size of database increases in the big data generation, data mining has become more important for making a scientific decision for business intelligence (Davenport and Harris, 2007).

Learning (or training) in data mining refers to identification of a functional relationship of the input variables and the target variable from a training dataset. There are two major paradigms of learning: supervised learning and unsupervised learning. In supervised learning, d–dimensional training data, $\mathcal{X} = \{\mathbf{x}_i | i = 1, ..., n,\ \mathbf{x}_i \in \mathbb{R}^d\}$, are given with corresponding labels or targets, $\mathcal{Y} = \{y_i | i = 1, ..., n,\ y_i = f(\mathbf{x}_i)\}$ (Kang, 2010). Labels are either -1 or 1 for binary classification problems while labels are continuous values for the regression problems. In unsupervised learning, only training data, $\mathcal{X}$, are given without their labels. The main goal of unsupervised learning is to understand the structure of the dataset. One

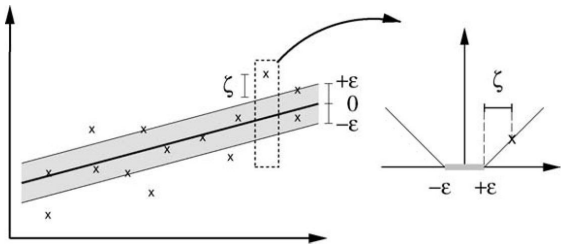


Figure 1.1: ε–tube based on the margin of training data and the ε–loss function of SVR (Smola and Schölkopf, 2002).

of the most popular approaches of unsupervised learning is the clustering analysis. Recently, Semi-Supervised Learning (SSL), training both labeled data and unlabeled data, has been widely researched in order to improve the generalization performance of the supervised learning.

1.1 Support Vector Regression

Support Vector Machine (SVM) was developed by Vapnik based on the Structural Risk Minimization (SRM) principle (Vapnik, 1995). Contrast to the conventional Empirical Risk Minimization (ERM) algorithms, SVM trains a dataset on direction of maximizing the generalization performance, not the empirical accuracy. Support Vector Classifier (SVC), a classification version of SVM, maximizes margins of training data, which is calculated by the distance between the closest data from different classes (Burges, 1998). Also, SVC has the capability of solving nonlinear problems using the kernel trick (Shawe-Taylor and Cristianini, 2000). With the great generalization performances, SVC has become one of the most spotlighted algorithms and has successfully applied to various areas, such as text categorization (Joachims, 1998), face recognition (Déniz et al., 2003), chemistry (Li et al., 2009), manufacturing (Widodo et al., 2007) and response modeling (Shin and Cho, 2006).

Support Vector Regression (SVR), a regression version of SVM, was proposed in order to solve nonlinear regression problems with a maximum margin algorithm (Smola and Schölkopf, 2002). SVR employs a ε–insensitive loss function (see Figure 1.1), and training data whose margin are less than ε are not counted as error. Hence, an ε–sized insensitive tube (ε–insensitive tube or ε–tube) has constructed while SVR training.

SVR has the same advantages of SVC. SVR also maximizes the generalization performance by employing the SRM principle with ε–insensitive loss function, and is capable of solving nonlinear problems with the kernel trick. With those advantages, SVR has been successfully applied to various areas: response modeling (Kim and Cho, 2012; Kim et al., 2008), virtual metrology (Kang et al., 2011), finance prediction (Pai and Lin, 2005), time–series prediction (Thissen et al., 2003) and environment (Ortiz-García et al., 2010).

Despite of the great generalization performance, there still exist open problems for SVR to overcome. In this dissertation, two major issues of SVR are studied:

(1) **Training complexity**: The training complexity of SVR is relatively high to analyze a large size of dataset, which makes SVR difficult to be applied to real–world datasets. An effort of reducing training complexity is needed to be studied.

(2) **Semi–supervised SVR (SS–SVR)**: With the importance of using unlabeled data, SSL has been widely studied. However SSL for SVR is rarely proposed and is possible to be improved. For SS–SVR, issues which should be considered are that (a) how to use the unlabeled data along with the labeled data, (b) how to train SVR with a large size of the unlabeled data.

1.2 Data Selection

One of the major drawbacks of SVR is the training complexity. The training complexity of SVR is strongly correlated to the number of training data, as is that of SVM: $O(n^3)$ of the training time complexity and $O(n^2)$ of the training memory complexity, where n is the number of training data. The training time of SVR is expensive, and occasionally, SVR does not work in a limited memory space for large datasets. Recently, the size of training dataset is getting larger and larger. Data analysis for a real–world problem includes the construction of various models with different samples of a dataset to verify multiple strategic actions. Moreover, SVR contains an additional hyper–parameter which requires that the SVC, ε, be set empirically. Hence, the training complexity problem is more critical for SVR than for SVC.

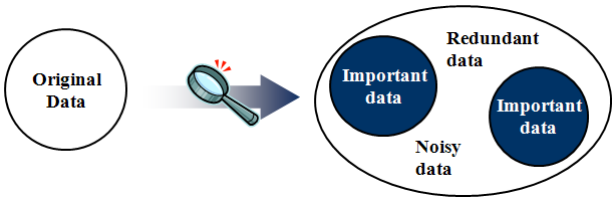


Figure 1.2: Important data among redundant and noisy data.

To overcome this training complexity problem, decomposition methods, such as Chunking, Sequential Minimal Optimization (SMO), SVM^{light}, Successive Overrelaxation (SOR) and Library SVM (LIBSVM), have been proposed in order to divide the original optimization problem into a series of smaller problems (Platt, 1998). However, the training complexities of these methods are still strongly correlated with the number of training data (Shin and Cho, 2007). Also, a Linear SVM method employing a cutting–plane algorithm (Joachims, 2006) and Primal Estimated sub–GrAdient SOlver for SVM (Pegasos) employing a stochastic gradient decent algorithm (Shalev-Shwartz et al., 2007) have been proposed. Both methods changed the optimization problem of the original SVM. However, the linear SVM and Pegasos were basically designed for modeling a spare dataset which including a large number of zero features and a small number of non–zero features. Moreover, they needs parameters to set empirically, and several iterations to converse.

Other research studies have focused on the data selection method, which relies on the assumption that there are important or informative data among the redundant data and noisy data in the original dataset (see Figure 1.2). The goal of data selection is that selection of important data (or removal of redundant and noisy data) to train before training all data. The performance loss occurred when removing training data should be minimized. At the same time, training time including data selection processing time should be shorter than the original training time.

For data selection for SVM, most data selection studies using methods such as SVM–KM (Almeida et al., 2000), Neighborhood Property based Pattern Selection (NPPS) (Shin and Cho, 2007), a cross–training based method (Bakir et al., 2005) have focused their efforts on classification problems, rather than regression problems. Those methods cannot be applied to regression problems. For regression problems, a Heuristic SVM (HSVM) (Wang and Xu, 2004), a k–NN based method (Sun and

Cho, 2006), ε–based data selection method (ε–DS) (Kim and Cho, 2006) and Reducing examples of SVR (RSVR) (Guo and Zhang, 2007) have been proposed. However, there have been some drawbacks with the use of these methods. HSVM is basically useful for time–series problems. Hence, an additional effort is needed to conduct the partitioning part of HSVM for non time–series problems. Moreover, HSVM and the k–NN based method have cut–off parameters that directly and empirically determine the number of data selected. ε–DS selects data inside the ε–DS while RSVR took an interesting idea which selects support vectors. However, HSVM, the k–NN based method, ε–DS and RSVR tend to degrade accuracy when it trains high dimensional datasets. A new data selection method for SVR is needed to be developed with the minimum accuracy loss and an easy usability.

1.3 Semi-Supervised Learning

The conventional supervised learning uses only labeled data to train. However, the labeled data are often difficult, expensive or time consuming to obtain. "SSL addresses this problem by using large amount of unlabeled data, together with the labeled data, to build better classifiers" (Zhu, 2006).

SSL is originally designed for classification problems. Co–training method (Blum and Mitchell, 1998; Mitchell, 1999), the most popular multi–view algorithm, is a common solution for SSL. Co–training constructs two different training models each of which trains different view of the labeled dataset. Graph–based method is one of other solutions for SSL, but the graph–based method is more likely to applied for transductive learning, which has a slightly different view than SSL (Joachims, 1999; Vapnik, 1995; Zhu, 2006). SSL and transductive learning are successfully applied to marketing (Lee et al., 2010), image sensing (Maulik and Chakraborty, 2011) and web mining (Sun et al., 2011) applications.

Recently, SSL for regression has been proposed (Brefeld et al., 2006; Cortes and Mohri, 2006; Sindhwani et al., 2005; Wang et al., 2011, 2010; Zhou and Li, 2007; Zhu, 2006). SSL for regression is more difficult than SSL for classification. In order to use the unlabeled data for training, SSL for regression needs to estimate continuous numbers as labels, which is more difficult than estimating binary labels for SSL for classification. Hence, most of those SSL for regression methods employ the multi–view algorithm

which estimates the labels of the unlabeled data using regression models as base learners. COREG (Zhou and Li, 2007), a co–training method for regression employing k–Nearest Neighbor (k-NN) regressor as the base learner, showed a great performance.

For the Semi–Supervised SVR (SS–SVR), co–training based method (Co–SVR) was proposed (Wang et al., 2011, 2010). Since the training performance of the base learner is very important for co–training, the idea that co–training employing SVR is reasonable. However, there are some drawbacks of Co–SVR. First of all, co–training is a time consuming method. Since co–training is an iterative method with adding the unlabeled data into training dataset, as the iterations go, the training complexity of co–training is getting higher. Since the training time complexity of SVR is $O(n^3)$, this problem is critical especially for Co–SVR. Second, co–training re–trains the base SVR model with adding one unlabeled data point independently, and measures the improvement of model accuracy. Since SVR employs SRM principle with ε–loss function to avoid overffiting, one added unlabeled data point may not affect to the model accuracy. Third, SVR estimates a regression function calculated by a linear combination of support vectors. Thus, the estimated labels of the unlabeled data are just interpolations of training data. In that case, the estimated labels of the unlabeled data do not give any new information of the underlying function. Finally, SVR estimates the regression function under the maximum margin algorithm with parameter of ε. The estimated labels of the unlabeled data are needed to be distributed around the regression function, not to be distributed on the function.

1.4 Contributions of this Dissertation

The main contributions of this dissertation are three–fold:

(1) A new data selection method, Margin based Data Selection (MDS), for SVR is proposed to reduce the training complexity of SVR. Because, the training complexity of SVR is highly related to the number of training data, the data selection is the most efficient approach to reduce the training complexity. For SVR, the most important data are support vectors, which affect the construction of a regression model. However, before training, there is no way to identify which training data will become the support vectors. Using the fact that support vectors are always located on or outside the ε–tube (See Figure 1.1),

the training data with a margin equal or greater than ε should be selected. With multiple bootstrap learning, the margins of all training data are estimated and data with a margin equal or greater than ε are selected. MDS automatically determines the number of data selected according to a parameter α which allows MDS to control the trade–off between the training complexity and model performance.

(2) A new data generation and selection framework for SS–SVR is proposed. For SSL, the key part is the use of the unlabeled data along with the labeled data. Probabilistic Local Reconstruction (PLR) (Lee et al., 2012a,b) is employed to estimate labels of the unlabeled data. PLR is a local topology based linear reconstruction method. Since the output of PLR is represented as a probabilistic form, the estimated label distribution of each unlabeled data point can be obtained. Then, the data generation step is conducted. In data generation step, training data are multiply generated from the estimated label distribution of the unlabeled data. The data generating rate is differed by the uncertainty of the estimation for each unlabeled data point. Since the unlabeled data are multiply generated, the number of training data is larger than the original training data including the unlabeled data. Hence, the data selection method, MDS is applied to select the important data for the training efficiency.

(3) Data selection for SVR and data generation and selection for SS-SVR are applied to real–world datasets. First, a response modeling dataset as a marketing problem is involved. In order to estimate the amount of money spent for each respondent, two–stage response modeling is proposed. MDS is applied to reduce the training complexity. Second, a virtual metrology dataset as a manufacturing problem is involved. With both real–world datasets, the possibility of the proposed methods applying to real–world problems is analyzed.

The remainder of this dissertation is organized as follows. In Chapter 2, literature reviews of SVR, data selection and SSL is presented. In Chapter 3, Voting based Data Selection (VDS), which is a preliminary version of MDS, and MDS algorithm is summarized as well as their experimental results on benchmark datasets. In Chapter 4, the data generation and selection method for SS–SVR is proposed with experimental results on benchmark datasets. In Chapter 5, the data selection method are applied to a real–world marking problem, i.e. response modeling. In Chapter 6,

experimental results for SS–SVR conducted on a real–world manufacturing problem is presented. Finally, Chapter 7 concludes this dissertation with the summary and limitations of the proposed approaches as well as the future works.

Chapter 2

Literature Review

2.1 Support Vector Regression

For a brief review of SVR, consider a regression function $f(\mathbf{x})$ to be estimated with training data $\{(\mathbf{x}_i, y_i), i = 1, \cdots, n\}$ as follows:

$$\begin{aligned} f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b, \quad \text{with} \quad \mathbf{w}, \mathbf{x} \in \mathbb{R}^d, b \in \mathbb{R} \\ \text{where} \quad \{(\mathbf{x}_1, y_1), \ldots (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \mathbb{R}. \end{aligned} \tag{2.1}$$

By the SRM principle, the generalization accuracy is optimized by the flatness of the regression function. Since the flatness is guaranteed on small $\mathbf{w}$, SVR is moved to minimize the norm, $\|\mathbf{w}\|^2$. An optimization problem could be formulated:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i - \mathbf{w} \cdot \mathbf{x}_i - b \leq \varepsilon \\ & \mathbf{w} \cdot \mathbf{x}_i + b - y_i \leq \varepsilon \end{aligned} \tag{2.2}$$

where ε is the size of the ε–tube. This is called the hard margin problem of SVR. In the hard margin problem, SVR does not allow any training data to be located outside the ε–insensitive tube (ε–tube). Since this assumption is too strong to solve real–world problems, the soft margin problem of SVR is formulated:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - \mathbf{w}\cdot\mathbf{x}_i - b \leq \varepsilon + \xi_i \\ & \mathbf{w}\cdot\mathbf{x}_i + b - y_i \leq \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \end{aligned} \tag{2.3}$$

where C, ε, and ξ_i, ξ_i^* are the trade–off cost between the empirical error and the flatness, the size of the ε–tube and slack variables, respectively. In the soft margin problem, the slack variables, ξ_i and ξ_i^*, are employed in order to robust to noisy data. The ε–loss function can be formulated:

$$\begin{aligned} |\xi|_\varepsilon &= 0 \qquad \text{if} \quad |\xi| \leqq \varepsilon \\ |\xi|_\varepsilon &= |\xi| - \varepsilon \quad \text{otherwise.} \end{aligned} \tag{2.4}$$

By introducing a dual set of variables α_i and α_i^*, the optimization problem could be converted into the unconstraint optimization problem:

$$\begin{aligned} L = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) - \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i^*\xi_i^*) \\ - \sum_{i=1}^{l}\alpha_i(\varepsilon + \xi_i - y_i + \mathbf{w}\cdot\mathbf{x}_i + b) \\ - \sum_{i=1}^{l}\alpha_i^*(\varepsilon + \xi_i^* - y_i - \mathbf{w}\cdot\mathbf{x}_i - b). \end{aligned} \tag{2.5}$$

With the partial derivatives of L, the dual optimization form can be obtained:

$$\begin{aligned} \text{Maximize} \quad & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)(\mathbf{x}_i\cdot\mathbf{x}_j) \\ & -\varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \;\text{ and }\; \alpha_i, \alpha_i^* \in [0, C]. \end{aligned} \tag{2.6}$$

Then, the solution of SVR can be obtained:

$$\mathbf{w} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\mathbf{x}_i, \text{ thus } f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)(\mathbf{x}_i \cdot \mathbf{x}_j) + b. \tag{2.7}$$

In order to solve nonlinear problems, kernel trick is employed. Let us the training data $\mathbf{x}_i$ be mapped into a high–dimensional feature space $\mathcal{F}$, $\Phi : \mathcal{X} \to \mathcal{F}$, where Φ is a mapping function. The optimization problem and the solution can be changed as:

$$\begin{aligned} \text{Minimize } & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{s.t. } & y_i - \mathbf{w} \cdot \Phi(\mathbf{x}_i) - b \leq \varepsilon + \xi_i \\ & \mathbf{w} \cdot \Phi(\mathbf{x}_i) + b - y_i \leq \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \end{aligned} \tag{2.8}$$

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) + b. \tag{2.9}$$

In order to calculate $\Phi(\mathbf{x})$, the explicit mapping function should be defined. However, if the calculation of the solution consists of the inner product of the mapped data, those inner product can be replaced by a kernel function by the kernel trick, $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. The polynomial kernel, $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$, the Gaussian kernel, $K(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2})$ and the sigmoid kernel $K(\mathbf{x}_i, \mathbf{x}_j) = tanh(\kappa(\mathbf{x}_i \cdot \mathbf{x}_j) + \Theta)$ are commonly used kernels (Schölkopf and Smola, 2002).

Except the bias term, b, the solution of SVR is constituted with a linear combination of training data and their weights, α_i and α_i^*. If either α_i or α_i^* is greater than zero, the data point $\mathbf{x}_i$ becomes a support vector. Support vectors are used to construct the regression function $f(\mathbf{x})$, while the non support vectors, which have zero α_i and α_i^* values, do not affect the results. Hence, support vectors are the most important and informative data for training SVR. With the ε–loss function, support vectors are always located outside the ε–tube.

2.2 Data Selection for Support Vector Regression

2.2.1 Time Complexity Reduction

Despite of the great generalization performance of SVR, one drawback that makes SVM difficult to use in practice is the training complexities. Nowadays, we have giga–tera–pita bytes of data to analyze in large databases. However, the training complexity of SVM is strongly correlated to the number of training data: $O(n^3)$ of the training time complexity and $O(n^2)$ of the training memory complexity, where n is the number of training data. Hence, the time it takes for SVM to train real world datasets is too long, and occasionally, SVM does not work because the size of the kernel matrix constructed from training data is too large for the memory limits. Moreover, in order to apply SVM to real–world problems, several sensitive hyper–parameters of SVM (or SVR) should be set, empirically.

To solve this drawback, some researches have been conducted using the decomposition method. The decomposition method splits an original optimization problem into a series of smaller optimization problems. Chunking, SMO, SVM^{light}, SOR and LIBSVM have been proposed with time complexity $T \cdot O(nq + q)$, where T is the number of iterations and q is the size of the working set (Platt, 1998). However, the training time complexities of those methods are still strongly correlated with the number of training data (Shin and Cho, 2007). Also, a Linear SVM method employing a cutting–plane algorithm (Joachims, 2006) and Primal Estimated sub–GrAdient SOlver for SVM (Pegasos) employing a stochastic gradient decent algorithm (Shalev-Shwartz et al., 2007) have been proposed. Both methods changed the optimization problem of the original SVM to reduce the training complexity of Quadratic Program (QP) in the optimization of SVM. The training complexity of the linear SVM method was $O(sn)$ while that of Pegasos was $O(\frac{d}{(\lambda\epsilon)})$, where s, d, λ and ϵ are the number of non–zero features, a bound of the number of non–zero features, a penalty parameter and a solution accuracy, respectively. However, the linear SVM and Pegasos were basically designed for modeling a spare dataset which including a large number of zero features and a small number of non–zero features. Moreover, they needs parameters to set empirically, and several iterations to converse. These parameters and iterations may be a critical issue when applying to real–world applications with a large number of training data.

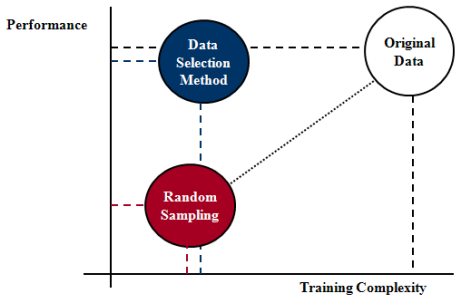


Figure 2.1: The goal of data selection method.

2.2.2 Data Selection Method

Other research studies focused on the data selection method. This method relies on the assumption that there are important or informative data among redundant data and noisy data in the original dataset. The goal of data selection is that selection of important data (or removal of redundant and noisy data) to train before training all data. The performance loss occurred when removing training data is needed to be minimized. At the same time, training time including data selection processing time should be shorter than the original training time (see Figure 2.1). In the data selection method, the most important data are selected, and these are used to form a smaller subset. For SVM, support vectors are the most important method. Hence, most researches of data selection for SVM have been focused on selecting support vectors. SVM–KM (Almeida et al., 2000), one of the first method proposed, employs k–Means clustering to check whether or not a data point is likely to become a support vector. SVM–KM runs k–Means clustering for all training data, and check whether or not each cluster consists more than one class of data. Clusters formed only by data that belong to the same class label can be disregard and used only their centers. On the other hand, clusters with more than one class are unchanged and trained. Neighborhood Property based Pattern Selection (NPPS) (Shin and Cho, 2007) made an improvement of the idea of SVM–KM by employing the k-Nearest Neighbors method (k-NN) to identify those data which are located near the decision boundary. NPPS identified data likely to become support vectors, when the data is located near the decision boundary but is not noise. With “Label Probability”, “Neighbors Entropy” and “Neighbors Match”, NPPS selected data likely to become

support vectors. Shin and Cho (2007) had also proposed a fast version of NPPS. Fast NPPS started k–NN with a small subset of all training data, and expanded the working subset. Fast NPPS improved the training speed of NPPS within a reasonable bound of accuracy. A Cross–Training based method selects a subset of data with an averaged margin calculated by the cross–training method (Bakir et al., 2005). The cross–training based method had proposed the possibility of estimating margin with sample set learning. However, those approaches were developed specifically for classification problems, not for ordinary regression problems.

2.2.3 Data Selection Method for Support Vector Regression

Recently, data selection methods designed especially for SVR such as a Heuristic SVM (HSVM) (Wang and Xu, 2004), ε–PS (Kim and Cho, 2008), Reducing examples of SVR (RSVR) (Guo and Zhang, 2007) and k–NN based method (Sun and Cho, 2006) have been proposed. The algorithm of HSVM is presented in Figure 2.2. HSVM splits the training data into k exclusive groups. Then the similarity is calculated between each data point and the center of each group as a reverse of their Euclidean distance as in Eq. 2.10. Data which have a greater similarity than the pre–fixed threshold are selected. The k–NN based data selection method was also proposed by taking entropy and variability into account. In some sense, HSVM reduces redundant data among the original dataset. The k–NN based method removes those data which are located in a dense region and have not a unique label value. After conducting k–NN for the training data, sparsity, variability and uniqueness can be calculated as in Eq. 2.11, Eq. 2.12 and Eq. 2.13, respectively. The algorithm of the stochastic version of this method is presented in Figure 2.3. This method has reduced the number of data maintaining a comparable level of accuracy. ε–DS is designed for selecting data falling inside of the ε–tube. The motivation of ε–DS is to remove noisy data and is to turn the soft margin problem of SVR into the hard margin problem. However, ε–DS tends to remove too many support vectors. The algorithm of ε–DS is presented in Figure 2.4. RSVR is a similar approaches to HSVM. However, RSVR employs an iterative search with k–NN while HSVM employs k–Means clustering. However, those four methods have a couple of critical parameters which are to be empirically determined: k for HSVM, k–NN based method and RSVR. Moreover the number of data selected was manipulated with parameters or thresholds

HSVM Algorithm

1. Initialize the similarity threshold S
2. Divide the training data into k groups of size m_j
3. For each data point group $D_j, j = 1, ...k$
   3.1. Compute the mean $\bar{\mathbf{x}}_j^{'}$ of D_j,
   and seek the center–like data point $\bar{\mathbf{x}}_j \in D_j$, which is closest to $\bar{\mathbf{x}}_j^{'}$
   3.2. Compute the similarity $S_i^j (i = 1, ...m)$
   between each data point $\mathbf{x}_i^j \in D_j$ and $\bar{\mathbf{x}}_j$ according to Eq. 2.10,
   where m_j is the number of training data belonging to D_j
   3.3. If $S_i^j > S$, then $\mathbf{x}_i^j$ is removed
4. Train SVR with the remained training data

Figure 2.2: The algorithm of HSVM.

to be determined by users without any guidelines. The number of data selected to train a model is rarely known for all real–world problems. Also, those methods tend to degrade accuracy when they train high dimensional datasets.

$$S(\mathbf{x}, \mathbf{y}) = f(\frac{1}{||\mathbf{x} - \mathbf{y}||_2}) = f(\frac{1}{\sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}}) \tag{2.10}$$

$$s_i = \frac{1}{k}\sum_{j=1}^{k} ||\mathbf{x}_i - \mathbf{x}_{NN(\mathbf{x}_i, j)}|| \tag{2.11}$$

$$v_i = \frac{1}{k}\sum_{j=1}^{k} |y_{NN(\mathbf{x}_i, j)} - \bar{y}_{NN(\mathbf{x}_i)}| \tag{2.12}$$

$$u_i = |y_i - \bar{y}_{NN(\mathbf{x}_i)}| \tag{2.13}$$

k–NN based method Algorithm (Stochastic)

1. Initialize the number of selected data, s
2. Compute $fitness(i)$ and $p(\mathbf{x}_i)$ for all $\mathbf{x}_i \in D$
   FOR(i=1 to n)
   $fitness(i) \leftarrow (2v_i + s_i)$
   END FOR
   FOR(i=1 to n)
   $p(\mathbf{x}_i) \leftarrow \frac{fitness(i)}{\sum_{j=1}^{n} fitness(j)}$
   END FOR
3. Select s data from D with $p(\mathbf{x}_i)$

Figure 2.3: The stochastic algorithm of the k–NN based method.

ε–DS Algorithm

1. Initialize the number of sample sets l
   Initialize the number of data in each sample set, m
   Initialize the number of data to be selected, s
2. Make l samples sets of size m, D_j for $j = 1, \cdots, l$, from the original dataset D by random sampling without replacement
3. Train SVR f_j with D_j, $\forall_j$
4. Evaluate the original dataset D by f_j, $\forall_j$
5. $M_{ij} = 1$, if a data point $\mathbf{x}_i$ is found inside the ε–tubes of f_j (otherwise $M_{ij} = 0$)
6. Calculate $p_i = \frac{\sum_{j=1}^{l} M_{ij}}{\sum_{i=1}^{n} \sum_{j=1}^{l} M_{ij}}$
7. Select s data stochastically without replacement based on p_i,
8. Train final SVR with s selected data

Figure 2.4: The algorithm of ε–DS.

2.3 Semi–Supervised Learning for Support Vector Regression

2.3.1 Semi–Supervised Learning

The conventional supervised learning uses labeled data to train. However the labeled data are often difficult, expensive or time consuming to obtain. Two major paradigms to use the unlabeled data with the labeled data are researched: SSL and transductive learning. The difference between SSL and the transductive learning is the perspective for the unlabeled data. In SSL, the unlabeled data are different from test data (Chapelle et al., 2006; Zhu, 2006). Hence, SSL constructs a model using both the labeled data and the unlabeled data, and then the constructed model is applied to estimate the labels of test data. On the other hand, in the transductive learning, the unlabeled data are same with the test data (Joachims, 1999; Vapnik, 1995). Hence, the concept of tranductive learning is to involving the unlabeled test data when training the model. The graph–based method is the most popular algorithm for transductive learning. The graph–based method defines a graph where the nodes are the labeled data and the unlabeled data in the dataset and the edges reflect the similarity based on local topologies of data (Zhu, 2006). The limitations for the graph–based methods are: (1) training complexity problem, (2) unstable to noisy data and (3) re–training is needed when the dataset has changed.

For SSL, co–training methods are widely proposed. Co–training (Blum and Mitchell, 1998; Mitchell, 1999) is a specific form of the multi–view method (de SA, 1993). Co–training trains two base models each of which has a different view for the training data from the other. The different views can be gained by different sampling, different model parameters or different features included. Each base model evaluates the unlabeled data and every unlabeled data have two labels from two base models. And then, each base model trains new training datasets adding unlabeled data one by one. An unlabeled data point which improves the model's accuracy most is added to the other model's training dataset. The basic idea of those co–training and the graph–based method is that the local topology of input variables play the key role in estimation of the unlabeled data.

2.3.2 Semi–Supervised Learning for Regression

Contrast to SSL for classification, SSL for regression is still an open problem. Only binary class labels are needed to be estimated for SSL classification. With the structure of input variables of the labeled data and the unlabeled data, the unlabeled data are determined whether their labels are -1 or +1 for SSL classification. However, continuous numbers are needed to be estimated to labels for SSL regression. Because of this obstacle, methods for SSL regression have mostly focused on co–training based algorithm. The advantage of co–training for SSL regression is that a regression model can be directly involved for estimating labels of the unlabeled data.

COREG (Zhou and Li, 2007) is one of the most well–developed algorithms for SSL regression. The algorithm of COREG is presented in Figure 2.5. In COREG, k–NN regression model is employed as the base models of co–training. If two base models train the same view of dataset, this method can be fallen into the self–training. In order to make a different view for both base models, two initial training datasets are constructed by sampling from the labeled data, randomly. In addition, each k–NN is set to use different parameter k, the number of nearest neighbors as well as the different distance measures: Mahalanobis distance and Euclidean distance. Plentiful experiments showed that COREG can improve the performance of the conventional supervised regression models or self–training based method.

Co–SVR (Wang et al., 2011, 2010) was also proposed. The algorithm of Co–SVR is presented in Figure 2.6. The framework of Co–SVR is very similar with that of COREG except Co–SVR employs SVR for base models. Even though Co–SVR is applied to a remote sensing water quality retrieving problem, there are some limitations for Co–SVR. Because SVR is a function estimation model, not a local topology based model, Co–SVR has a possibility to be trained as a self–training. Hence, the additional unlabeled data may not have new information for the underlying function.

To avoid the self–training problem, a local topology based method seems to be a better solution for SSL regression. In this dissertation, PLR is adapted as base learners. One of the basic local topology based method is the k–NN method. For a given test data $\mathbf{x}_*$, k–NN learning estimates its target value using the following equation,

$$\hat{y}_* = \sum_{i=1}^{k} w_{NN(\mathbf{x}_*,i)} y_{NN(\mathbf{x}_*,i)} = \mathbf{w}^T_{NN(\mathbf{x}_*)} \mathbf{y}_{NN(\mathbf{x}_*)}, \tag{2.14}$$

ALGORITHM: COREG

INPUT: labeled example set L, unlabeled example set U,
maximum number of learning iterations T,
number of nearest neighbors k_1, k_2
distance metrics D_1, D_2

PROCESS:
$L_1 \leftarrow L$; $L_2 \leftarrow L$
Create pool U' of size s by randomly picking examples from U
$h_1 \leftarrow kNN(L_1, k_1, D_1)$; $h_2 \leftarrow kNN(L_2, k_2, D_2)$
Repeat for T rounds:
for $j \in \{1, 2\}$ **do**
for each $\mathbf{x}_u \in U'$ **do**
$\Omega_u \leftarrow Neighbors\,(\mathbf{x}_u, L_j, k_j, D_j)$
$\hat{\mathbf{y}}_u \leftarrow h_j(\mathbf{x}_u)$
$h'_j \leftarrow kNN(L_j \cup \{(\mathbf{x}_u, \hat{\mathbf{y}}_u)\}, k_j, D_j)$
$\delta_{\mathbf{x}_u} \leftarrow \sum_{\mathbf{x}_i \in \Omega_u} \left((\mathbf{y}_i - h_j(\mathbf{x}_i))^2 - (\mathbf{y}_i - h'_j(\mathbf{x}_i))^2 \right)$
end of for
if there exists an $\delta_{\mathbf{x}_u} > 0$
then $\tilde{\mathbf{x}}_j \leftarrow \arg\max_{\mathbf{x}_u \in U'} \delta_{\mathbf{x}_u}$; $\tilde{\mathbf{y}}_j \leftarrow h_j(\tilde{\mathbf{x}}_j)$
$\pi_j \leftarrow \{(\tilde{\mathbf{x}}_j, \tilde{\mathbf{y}}_j)\}$; $U' \leftarrow U' - \{\tilde{\mathbf{x}}_j\}$
else $\pi_j \leftarrow \emptyset$
end of for
$L_1 \leftarrow L_1 \cup \pi_2$; $L_2 \leftarrow L_2 \cup \pi_1$
if neither of L_1 and L_2 changes **then exit**
else
$h_1 \leftarrow kNN(L_1, k_1, D_1)$; $h_2 \leftarrow kNN(L_2, k_2, D_2)$
Replenish U' to size s by randomly picking examples from U
end of Repeat
$f_1 \leftarrow Regressor(L_1)$; $f_2 \leftarrow Regressor(L_2)$

OUTPUT: regressor $f^*(\mathbf{x}) \leftarrow \frac{1}{2}(f_1(\mathbf{x}) + f_2(\mathbf{x}))$

Figure 2.5: The algorithm of COREG.

where $\mathrm{NN}(\mathbf{x}_*, i)$ is the ith nearest neighbor for $\mathbf{x}_*$. This equation has two user-specific parameters to be empirically determined: (1) the number of nearest neighbors, k and (2) the weights assigned to the selected neighbors, $\mathbf{w}_{NN(\mathbf{x}_*)} = [w_{NN(\mathbf{x}_*,1)}, ..., w_{NN(\mathbf{x}_*,k)}]^T$. (Kang and Cho, 2008; Lee et al., 2012b)

With the assumption that a data point in the input space can be described by a combination of its neighbors, the local reconstruction is the problem to capture the underlying combination or topology to minimize the difference between the data point and its description (Kang and Cho, 2008). The Locally Linear Reconstruction (LLR) method attempts to describe a given test data point $\mathbf{x}_*$ by the linear reconstruction of its k nearest neighbors:

1. Produce initial training set $L1$, $L2$ for SVR $h1$ and $h2$ from the labeled example set L. Prepare initial unlabeled example set $U1$, $U2$ for $h1$ and $h2$ from the unlabeled example set U.
2. Use GA and labeled sample set $L1$, $L2$ to choose the parameters for the two SVRs respectively.
3. Regressor $h1$ estimates unlabeled example set $U1$, choose the most confidently labeled example and its estimation result join to the training set $L2$ of $h2$.
4. Regressor $h2$ estimates unlabeled example set $U2$, choose the most confidently labeled example and its estimation result join to the training set $L1$ of $h1$.
5. Update $L1$, $U1$, $L2$, $U2$, then retraining regressors $h1, h2$.
6. If the maximum number of iterations has not reached, go to step 3, else continue.
7. Use regressor $h1$, $h2$ to predict for new samples, the final regression result is the mean value of the two regressors' outputs.

Figure 2.6: The algorithm of Co–SVR.

$$\mathbf{x}_* = \mathbf{X}_{NN}\mathbf{w}_{NN}, \tag{2.15}$$

where $\mathbf{x}_{NN(i)}$, $i = 1, ..., k$ and $\mathbf{x}_*$ and $\mathbf{X}_{NN} = [\mathbf{x}_{NN(1)}, ..., \mathbf{x}_{NN(k)}]$ denote the ith nearest neighbor of a test data point and neighbors matrix of the $\mathbf{x}_*$. The LLR can be solved by minimizing the reconstruction error $E(\mathbf{w}_{NN})$ as follows,

$$E(\mathbf{w}_{NN}) = \frac{1}{2}(\mathbf{x}_* - \mathbf{X}_{NN}\mathbf{w}_{NN})^T(\mathbf{x}_* - \mathbf{X}_{NN}\mathbf{w}_{NN}). \tag{2.16}$$

The explicit solution of the LLR regression is

$$\mathbf{w}_{NN} = [(\mathbf{x}_*^T\mathbf{X}_{NN})(\mathbf{X}_{NN}^T\mathbf{X}_{NN})^{-1}]^T. \tag{2.17}$$

PLR is a general and probabilistic form of LLR. PLR employs a probabilistic view in order to capture the reconstruction uncertainty. Hence, the reconstruction equation can be re–formed by adding the ϵ which is a uncertainty of the reconstruction as follow:

$$\mathbf{x}_* = \mathbf{X}_{NN}\mathbf{w} + \epsilon, \quad \epsilon \sim N(\mathbf{0}, \sigma^2\mathbf{I}). \tag{2.18}$$

Then, the likelihood is the test data point $\mathbf{x}_*$ is

$$p(\mathbf{x}_*|\mathbf{X}_{NN}, \mathbf{w}) = N(\mathbf{x}_*|\mathbf{X}_{NN}\mathbf{w}, \sigma^2), \tag{2.19}$$

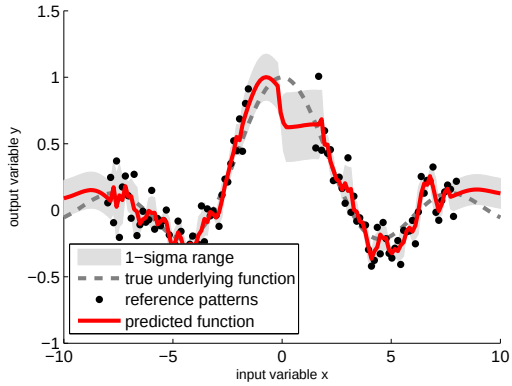


Figure 2.7: The predicted target variances of PLR regression with the RBF kernel.

with a zero mean Gaussian prior over the weight parameters, $p(\mathbf{w}) = N(\mathbf{w}|\mathbf{0}, \Sigma)$, the posterior of the weights can be calculated using Bayes' rule as follows,

$$\begin{aligned} p(\mathbf{w}|\mathbf{X}_{NN}, \mathbf{x}_*) \quad &\propto p(\mathbf{x}_*|\mathbf{X}_{NN}, \mathbf{w})p(\mathbf{w}) \\ &= N(\mathbf{x}_*|\mathbf{X}_{NN}\mathbf{w}, \sigma^2)N(\mathbf{w}|\mathbf{0}, \Sigma) \\ &= N(\mathbf{w}|\mu_p, \Sigma_p), \end{aligned} \tag{2.20}$$

where the posterior mean $\mu_p = \sigma^{-2}(\sigma^{-2}\mathbf{X}_{NN}^T\mathbf{X}_{NN} + \Sigma^{-1})^{-1}\mathbf{X}_{NN}^T\mathbf{x}_*$, which is the weight solution in PLR, and the posterior covariance matrix $\Sigma_p = (\sigma^{-2}\mathbf{X}_{NN}^T\mathbf{X}_{NN} + \Sigma^{-1})^{-1}$.

In order to solve the nonlinear problems, the kernel trick can be employed for PLR. The final predictive distribution for a given test data point $\mathbf{x}_*$ with kernelizing is,

$$\begin{aligned} y_* \quad &= \mu_p^T(\mathbf{y}_{NN} - \bar{y}_{NN}\mathbf{1}_{k\times 1}) + \bar{y}_{NN} \\ &= \mathbf{k}_*^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}_{NN} + \bar{y}_{NN}[1 - \mathbf{k}_*^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{1}_{k\times 1}], \end{aligned} \tag{2.21}$$

$$\sigma_*^2 = \frac{1}{k}(\mathbf{y}_{NN} - \bar{y}_{NN}\mathbf{1}_{k\times 1})^T(\sigma^{-2}\mathbf{K} + \mathbf{I})^{-1}(\mathbf{y}_{NN} - \bar{y}_{NN}\mathbf{1}_{k\times 1}), \tag{2.22}$$

where $\mathbf{1}_{k\times 1}$ is the $k \times 1$ column vector with every element being one. Then the predictive variance of the test data point $\mathbf{x}_*$ can be obtained as:

$$\sigma_{NN}^2 = \frac{1}{k}(\mathbf{y}_{NN} - \bar{y}_{NN}\mathbf{1}_{k\times 1})^T(\mathbf{y}_{NN} - \bar{y}_{NN}\mathbf{1}_{k\times 1}). \tag{2.23}$$

The PLR solution and the predicted target variance for given test data is illustrated in Figure 2.7

Chapter 3

Data Selection for Support Vector Regression

In Chapter 3, the algorithms of the data selection methods for SVR are presented as well as their experimental results. In order to reduce the training complexity of SVR, the selection of a smaller subset consisting important data to train is one of the effective approaches. For SVR, the most important data are support vectors. Only support vectors are involved in the calculation of the regression function while others are not used at all. Hence, SVR can train the same regression model even if input data were only support vectors. For the data selection's view, the support vectors are an ideal subset to be selected. Unfortunately, there is no way to identify support vectors before training all data. The goal of the proposed method is to find support vectors without training all data.

For the data selection of SVR, two data selection algorithms are proposed. First, Voting based Data Selection (VDS) is proposed. VDS is an early version of data selection for SVR research, which shows the possibility of selecting data which are likely to become support vectors by binary voting of multiple sample learning. Second, Margin based Data Selection (MDS) is proposed. MDS is an extended and general version of VDS. MDS estimates margin of all training data with multiple sample learning. And then, MDS selects data which are likely to become support vectors by comparing the margin and the ε. The purpose of both VDS and MDS is to select data which are likely to become support vectors in a short time. The notation and parameters used in Chapter 3 is presented in Figure 3.1.

Notations

D: Original dataset

D_j: The jth sample set from D

f_j: Trained SVR from the jth sample set

R_{ij}: Regression error of a data point $\mathbf{x}_i$ evaluated by f_j

M: Marking matrix for VDS

L_i: Likelihood score of a data point $\mathbf{x}_i$ for VDS

S: The expected number of support vectors for VDS

Margin($\mathbf{x}_i$): The averaged margin of data point $\mathbf{x}_i$ for MDS

Parameters

l: The number of sample sets

m: The number of data in a sample set (m% of the original data)

α: The accuracy–training time control parameter for MDS

Figure 3.1: Notations and parameters for VDS and MDS.

3.1 Voting based Data Selection

VDS is a simple version of data selection for SVR (Kim and Cho, 2008). The geometrical characteristic of support vectors are used to identify data which are likely to become support vectors. Support vectors are located on or outside the ε–tube. Hence, training data which are likely to become support vectors are probably located on or outside the ε–tube. Thus, the proposed data selection method can be summarized as estimation of the regression errors (residuals) of training data. Since the process time of the data selection should be shorter than the training time of all data, multiple sample learning is employed.

Initially, l bootstrap samples $D_j = \{(\mathbf{x}_i^j, y_i^j), \quad i = 1, \cdots, m\% \; of \; n \;$ and $j = 1, \cdots l\}$ containing m% of the original training data from the original dataset $D = \{(\mathbf{x}_i, y_i), i = 1, \cdots, n\}$ is constructed. Then, an SVR is trained with each sample set D_j and obtained l SVR regression functions f_j ($j = 1, \cdots, l$), which may not perform well but the training complexity is much lower than that of the original. Every data $\mathbf{x}_i$ in the original dataset D is evaluated by each regression function f_j, to identify whether it is located inside or outside the ε–tube. If a data point $\mathbf{x}_i$ is located on or outside the ε–tube of f_j, the data point is marked as 1, i.e. $M_{ij} = 1$. Then $L_i = \sum_{j=1}^{l} M_{ij}$, the number of total markings of the data point $\mathbf{x}_i$ can

VDS Algorithm

1. Initialize the number of sample sets l
   Initialize the number of data in each sample set, m
2. Make l samples sets of size m, D_j for $j = 1, \cdots, l$, from the original dataset D by random sampling without replacement
3. Train SVR f_j with D_j, $\forall_j$
4. Evaluate the original dataset D by f_j, $\forall_j$
5. $M_{ij} = 1$, if a data point $\mathbf{x}_i$ is found outside the ε–tubes of f_j (otherwise $M_{ij} = 0$)
6. Calculate $L_i = \sum_{j=1}^{k} M_{ij}$
7. Calculate $S = \frac{1}{k} \sum_{j=1}^{k} s_j$, where $s_j = \sum_{i=1}^{n} m_{ij}$
8. Select S data deterministically with largest L_i,
   or select S data stochastically without replacement according to $p_i = \frac{L_i}{\sum_{i=1}^{n} L_i}$.
9. Train final SVR with S selected data

Figure 3.2: The algorithm of VDS.

be calculated, which is used as the estimated likelihood of $\mathbf{x}_i$ to become a support vector. At the same time, the expected number of support vectors $S = \frac{1}{l} \sum_{j=1}^{l} s_j$, (where $s_j = \sum_{i=1}^{n} M_{ij}$) can be calculated by averaging the number of data marked by f_j. Finally, VDS select S data deterministically with largest L_i. Or, VDS select S data stochastically based on the probability $p_i = \frac{L_i}{\sum_{i=1}^{n} L_i}$. Finally, an SVR is trained again with the selected data. Figure 3.2 presents the algorithm. One of the advantages of VDS is that the critical parameter which affects the number of data selected is not involved.

3.2 Margin based Data Selection

VDS estimates the likelihood of becoming a support vector with binary voting. Even this method was effective, information loss can occur. During the data selection steps, the regression errors (residuals) of all training data are calculated. Then each residual value is turned into a binary value which only explained whether or not each data point was located outside the ε–tube. Information was lost by discarding the residual values, which had more information than the binary values. In addition, this method did not have a parameter which allowed users to control the accuracy–training

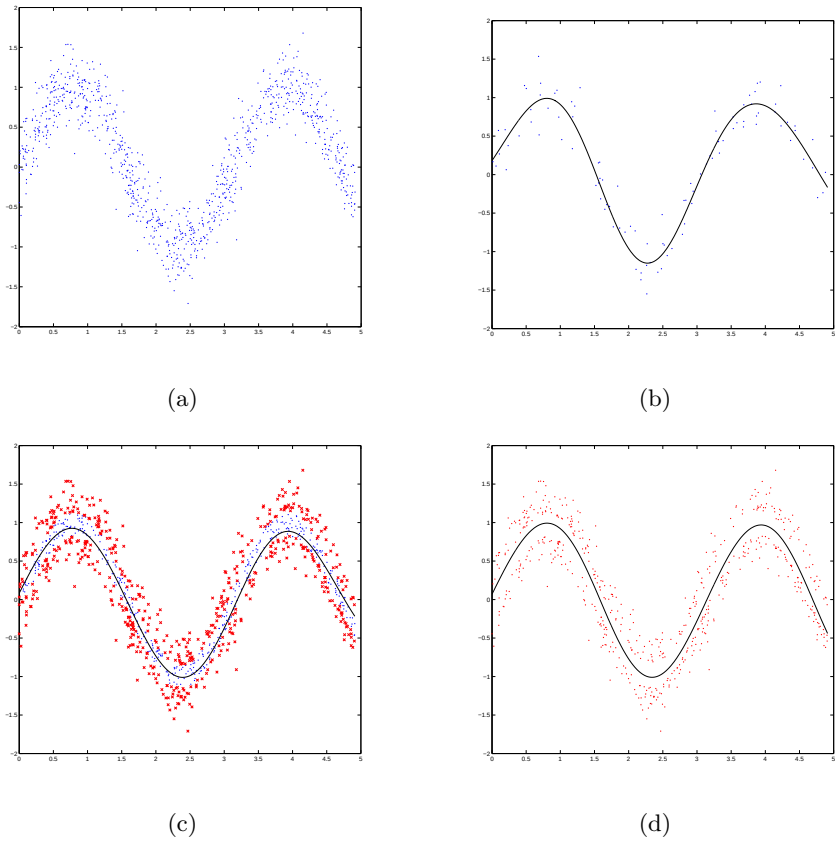


Figure 3.3: A graphical example of MDS.

time trade off.

MDS is an extended and general version of VDS. First of all, MDS estimates margin of each training data using the residual values, directly. In the SVR principle, the distance between a data point and the regression function is called the margin of the data point. Using multiple sample learning, MDS estimates margin for all training data and selects data with a margin greater than ε. Hence, the information loss occurred by the binary voting of VDS can be minimized. Also, MDS employs a parameter which controls the accuracy–training time trade off.

The early part of MDS is same as VDS. Initially, l bootstrap samples $D_j = \{(\mathbf{x}_i^j, y_i^j),\ \ i = 1, \cdots, m\%\ of\ n\ \text{ and }\ j = 1, \cdots l\}$ containing $m\%$

of the original training data from the original dataset $D = \{(\mathbf{x}_i, y_i), i = 1, \cdots, n\}$ is constructed. Then, an SVR is trained with each sample set D_j and obtained l SVR regression functions f_j $(j = 1, \cdots, l)$, which may not perform well but the training complexity is much lower than that of the original. Every data $\mathbf{x}_i$ in the original dataset D is evaluated by each regression function f_j. Then, the regression error of the data point $\mathbf{x}_i$ evaluated by f_j as R_{ij} can be recorded. Using R_{ij}, the averaged margin of data point $\mathbf{x}_i$, Margin($\mathbf{x}_i$), can be calculated with following equation:

$$\text{Margin}(\mathbf{x}_i) = \text{Average}_j(R_{ij}) + \alpha \times \text{Std}_j(R_{ij}) \quad \text{where} \;\; 0 \leq \alpha \leq 1, \tag{3.1}$$

where α is a control parameter of the accuracy–training time trade–off. Figure 3.3 illustrates an example of MDS: (a) original dataset and an SVR trained on it, (b) a sample set and an SVR trained on it, (c) original dataset and data outside the ε–tube marked as red and the others as blue (d) selected data and the resulting SVR trained using them.

If $\alpha = 0$, the average value of R_{ij} over j is used as the margin of data point $\mathbf{x}_i$. However, if $\alpha > 0$, MDS accounts for the standard deviation of R_{ij} over j. Hence, the parameter α controls the selection sensitivity, which determines the number of data selected by MDS. α has some advantages over the threshold parameters of other algorithms. First, α does not directly determine the number of data selected. Rather, α plays a role in determining how much additional information is added to the baseline $\text{Average}_j(R_{ij})$. Additionally, from the experimental results in Section 3.3, α is bounded in $0 \sim 1$.

The data selection rule compares Margin($\mathbf{x}_i$) and ε, a hyper-parameter of SVR, as depicted in Eq. 3.2. If Margin($\mathbf{x}_i$) is greater than or equal to ε, the data point $\mathbf{x}_i$ is estimated to be located outside the ε–tube and is likely to be an SV. If Margin($\mathbf{x}_i$) is smaller than ε, $\mathbf{x}_i$ is located inside the ε–tube, which means $\mathbf{x}_i$ is not likely to be an SV. Hence, we can say that MDS automatically determines the number of data selected based on the averaged margin and the pre–defined parameter ε.

$$\text{Margin}(\mathbf{x}_i) \geq \varepsilon \tag{3.2}$$

In VDS, however, the margin of a data point $\mathbf{x}_i$ was not estimated. Rather, whether or not a data point $\mathbf{x}_i$ was located outside the ε–tube

MDS Algorithm

1. Initialize the number of sample sets, l
   Initialize the number of data in each sample set, m
   Initialize the control parameter, α
2. Make l sample sets of size m, D_j for $j = 1, \cdots, l$, from D
   by random sampling without replacement
3. Train SVR f_j with D_j, $\forall_j$
4. Evaluate the original dataset D by f_j, $\forall_j$
5. R_{ij} = Regression error of $\mathbf{x}_i$ evaluated by f_j
6. Margin($\mathbf{x}_i$) = Average$_j(R_{ij})$ + $\alpha\times$ Std$_j(R_{ij})$
7. If Margin($\mathbf{x}_i$) $\geq \varepsilon$, select the data point $\mathbf{x}_i$
8. Train final SVR with S selected data

Figure 3.4: The algorithm of MDS.

was only checked. Thus, important information in R_{ij} was discarded by converting R_{ij} into M_{ij} as in Eq. 3.3. Moreover, VDS work only provided the likelihood score of a data point $\mathbf{x}_i$ to become an support vectors, while MDS gives us the margin of a data point $\mathbf{x}_i$. Also, the parameter of MDS, α, which controls the sensitivity of the number of data selected, results in an MDS different from that of VDS. MDS also has the advantage that the number of data selected is determined automatically.

$$\begin{aligned} &\text{If } R_{ij} \geq \varepsilon, \text{ then } M_{ij} = 1 \\ &\text{If } R_{ij} < \varepsilon, \text{ then } M_{ij} = 0 \end{aligned} \tag{3.3}$$

The MDS algorithm is presented in Figure 3.4.

3.3 Experimental Results

3.3.1 Experiment Setting

A total of 20 datasets, including two artificial datasets and 18 real world benchmark datasets, were used for the experiments. Real world benchmark

Table 3.1: Datasets used in the experiments for the data selection

No.	Name	♯ Train	♯ Test	♯ Attribute	Origin	Feature
1	Artificial Dataset	1000	1000	1	Generated	Art.
2	Add10	2000	2000	5	Delve Datasets	Art.
3	Santa Fe A	890	100	10	Santa Fe Comp.	T.S.
4	Santa Fe D	2000	2000	10	Santa Fe Comp.	T.S.
5	Santa Fe E	1490	500	10	Santa Fe Comp.	T.S.
6	Sun Spot	2000	1000	10	TSDL	T.S.
7	Melbourne Temp.	2000	1000	10	TSDL	T.S.
8	Gold	700	300	10	TSDL	T.S.
9	S&P 500	2000	1000	10	TSDL	T.S.
10	Wind	2000	2000	11	Statlib	Non–T.S.
11	Abalone	2000	2000	10	Delve Datasets	Non–T.S.
12	Computer Activity	2000	2000	12	Delve Datasets	Non–T.S.
13	Bank 8FM	2000	2000	8	Delve Datasets	Non–T.S.
14	Bank 8NH	2000	2000	8	Delve Datasets	Non–T.S.
15	Pumadyn 8FM	2000	2000	8	Delve Datasets	Non–T.S.
16	Pumadyn 8NH	2000	2000	8	Delve Datasets	Non–T.S.
17	Census House 8L	2000	2000	8	Delve Datasets	Non–T.S.
18	Census House 8H	2000	2000	8	Delve Datasets	Non–T.S.
19	Census House 16L	2000	2000	16	Delve Datasets	Non–T.S.
20	Census House 16H	2000	2000	16	Delve Datasets	Non–T.S.

datasets were gathered from Delve datasets[1], Time Series Data Library (TSDL)[2] and Statlib[3]. All datasets are summarized in Table 3.1.

The features of regression datasets can be partitioned into three types. "Art.", "T.S." and "Non–T.S." indicates an artificial dataset, a time series dataset and a non–time series multi–variate dataset, respectively. An artificial dataset, D1, was generated based on the mathematical function, $y = \sin(2x) + \xi$ where $x \in [0, 5]$ and $\xi \sim N(0, 0.5^2)$. Add10 dataset is another artificial dataset gathered from the Delve datasets. only five relevant input features were included, excluding five noise terms. Time series datasets were reformulated as regression problems by using the previous 10 values to estimate the following single value, which is a typical way to solve time series problems. The Wind dataset was reformulated to estimate the wind speed of the Dublin station using other 11 observed stations' wind speeds. The dataset 13 to the dataset 20 from Table 3.1 came from three datasets: Bank, Pumadyn and Census House. The number following the dataset name denotes the number of features used. FM, NH, L and H stand for 'fairly linear–moderate noise', 'nonlinear–high noise',

[1]Delve Dataset: `http://www.cs.toronto.edu/~delve/data/datasets.html/`

[2]TSDL: `http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/`

[3]Statlib: `http://lib.stat.cmu.edu/datasets/`

'low task difficulty' and 'high task difficulty', respectively. We expected to analyze the model performances varied with the different characteristics of training datasets, such as linearity and training complexity. To evaluate the performances, the original dataset was randomly split into training data and test data. The hyper–parameters of SVR were determined by cross–validation with C $\times$ $\varepsilon = \{0.1, 0.5, 1, 3, 5, 7, 10, 20, 50, 100\} \times \{0.01, 0.05, 0.07, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9, 1\}$. RBF kernel was used as a kernel function and the kernel parameter σ was fixed to 1.0 for all datasets. All datasets were normalized.

VDS and MDS were compared with HSVM, k–NN based method (Sun and Cho, 2006) and random sampling. For the HSVM method, the partitioning parameter is set to 10 while the similarity threshold was fixed at 1.2. Since, HSVM is designed only for time series problems, the partitioning step was exchanged with k–Means clustering to be used for non–time series problems. A stochastic version of the k–NN based method was implemented. The k of k–NN was fixed to 5. To use the k–NN based method, the number of data selected should be determined as a parameter. The parameter is set to be similar to the number of data selected by MDS. VDS and MDS have two parameters to set. Based on the model parameter selection which is presented in Section 3.4, l and m were determined to be 10 and 10% of n, respectively. The performances of each method are measured by Root Mean Squared Error (RMSE) (Eq. 3.4) and training time (sec.), including data selection time and SVR training time. All experimental results were averaged over 30 repetitions.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2} \tag{3.4}$$

3.3.2 Experimental Results

Figure 3.5 shows the experimental results of the artificial dataset, Add10 dataset, Sunspot dataset and Melbourne temperature dataset. The pairs of RMSE and training time in seconds are plotted corresponding to each method. The closer a result is plotted to the origin, the better the method performs. The solid line with triangles indicates the results of the random sampling from 10% to 100% of n. Marked squares indicate the experimental results of MDS with α=0, α=0.5 and α=1, respectively. The results of random samples were almost polynomially decreased as the number of

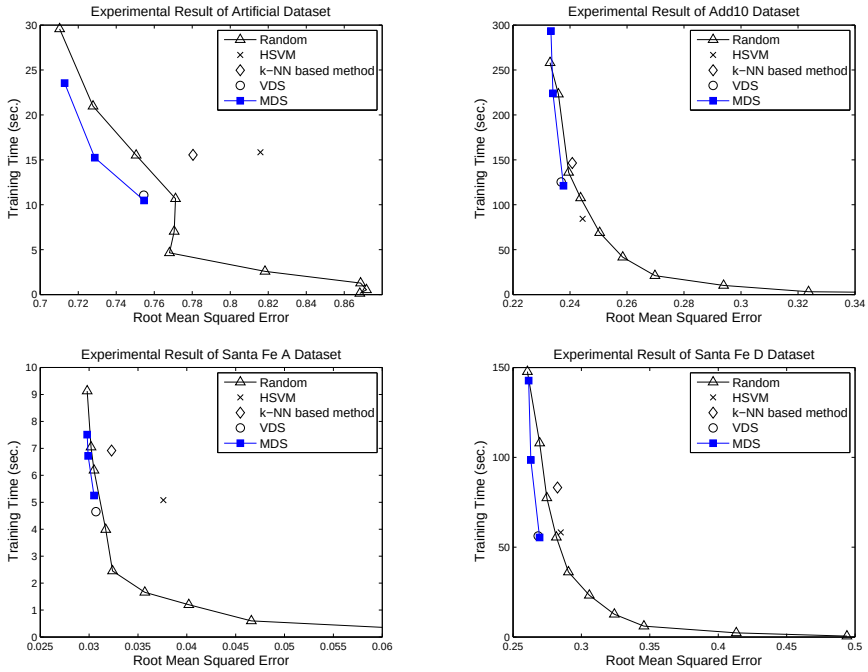


Figure 3.5: Experimental results of Dataset 1 to Dataset 4.

data selected decreases. As shown in Figure 3.5, the result of the MDS are satisfactory. MDS preformed better than benchmark methods, including random sampling in terms of pairs of RMSE and training time.

Figure 3.6 and Figure 3.7 show the rest of the experimental results. MDS showed more accurate regression performances than the random sampling given the same training time. MPBS showed different RMSE and training time pairs with the different values of α. It seems that α can control the accuracy of the result and the training time complexity. MDS outperformed the benchmark methods the datasets, regardless of whether the dataset were artificial, time–series or non–time series. As mentioned earlier, Dataset 13 to Dataset 20 have their own characteristics. In Figure 3.7, the figures presented in the left column are the results from datasets that have relatively little with linear noise, while the figures in the right column are the results from datasets with relatively large nonlinear noise. MDS works well over all kinds of datasets, regardless of noise level and the linearity of the noise.

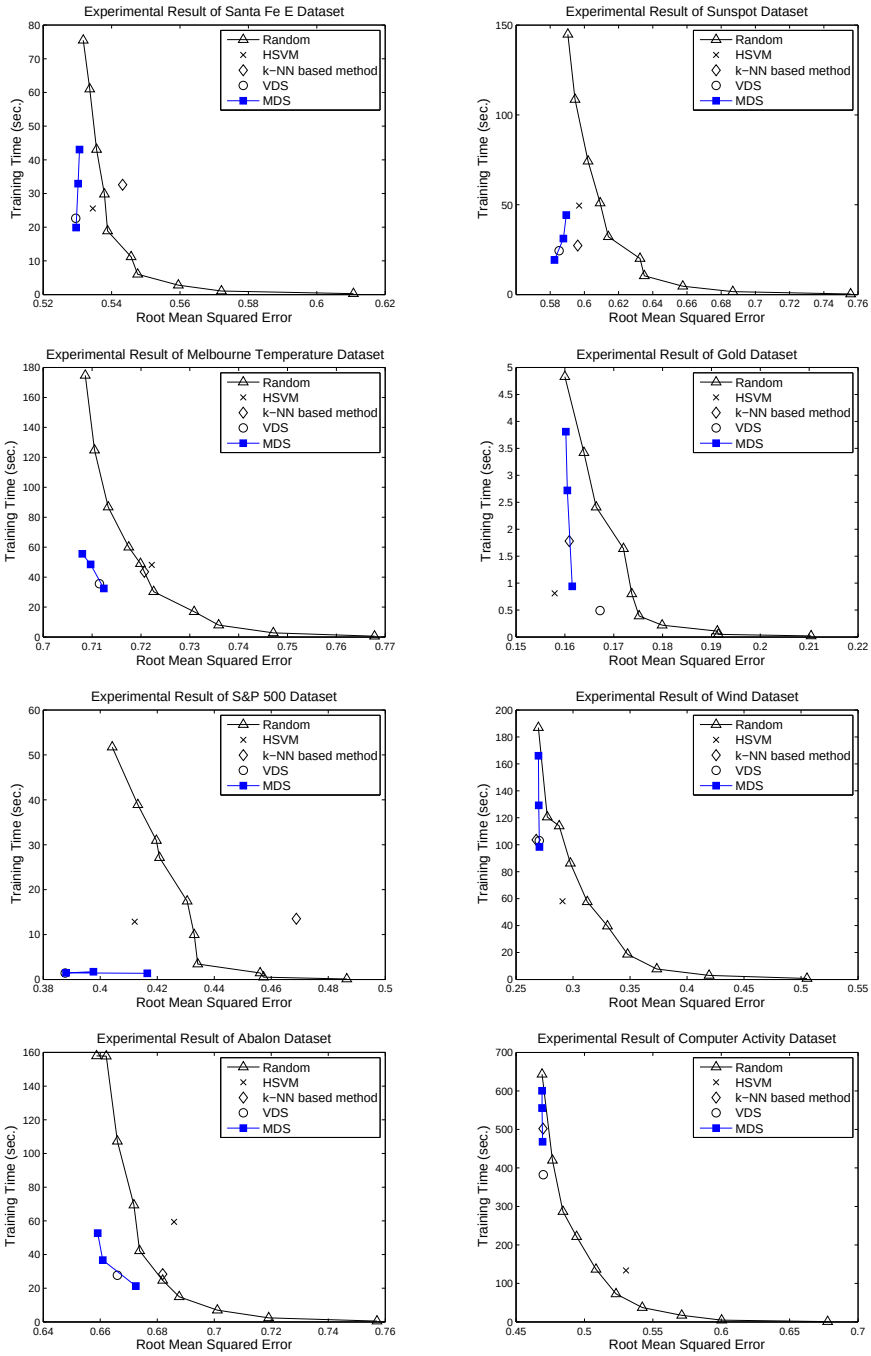


Figure 3.6: Experimental results of Dataset 5 to Dataset 12.

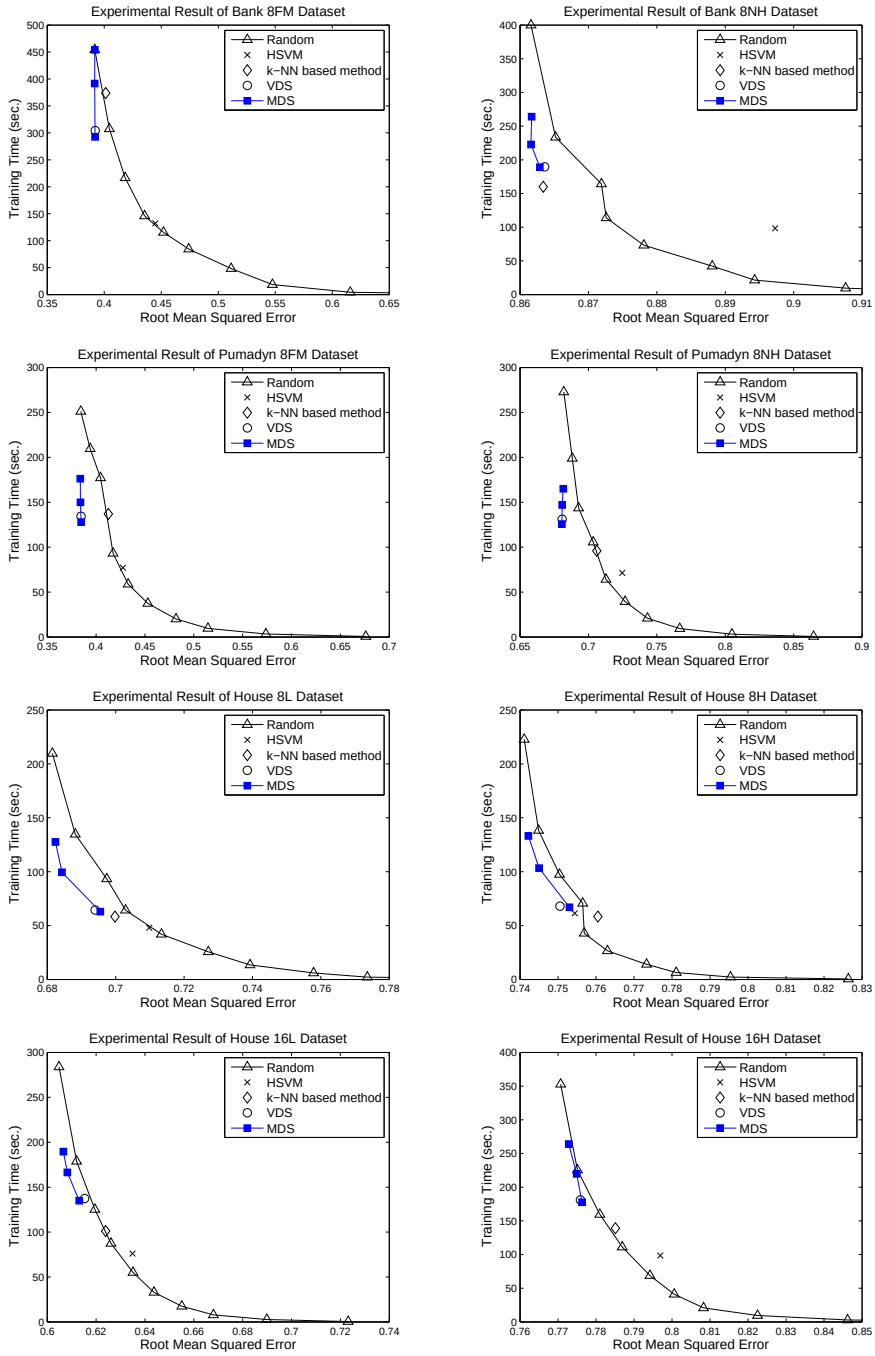


Figure 3.7: Experimental results of Dataset 13 to Dataset 20.

Table 3.2: Training time ratio, evaluation time ratio and RMSE ratio of bagging compared to MDS on percentage.

	MDS	Bagging, 10%	Bagging, 30%	Bagging, 50%	Bagging, 70%
Training time ratio	100	14.65	88.23	187.76	326.94
Evaluation time ratio	100	179.65	441.35	640.80	836.46
RMSE ratio	100	142.94	113.78	107.57	104.44

Since MDS employed a multiple sample set based learning, the comparison of MDS with Bootstrap Aggregating (Bagging) (Breiman, 1996) for SVR was conducted. Bagging is one of the finest bootstrap sample learning method. The original data are partitioned to l bootstrap sample set with replacement. In each bootstrap sample set, the number of data is smaller than that of the original data. Then, the base learners train each bootstrap sample set. The final decision for a test data point is obtained by merging (voting or averaging) the results of all base learners. The base learners of bagging should be low bias and high variance model, and the variance of base learners is smoothed by merging. For bagging, the hyper–parameters for SVR were set to the same values of those for MDS. The number of bootstraps of bagging was set to 10, which was same as the default value of l for MDS. The number of data in a bootstrap was varied from 10% of the original data to 30%, 50% and 70%.

Figure 3.8 illustrates the experimental results on 20 datasets. Figure 3.8 (a) depicts the training time ratio of bagging SVR compared to MDS on percentage while Figure 3.8 (b) and Figure 3.8 (c) depict the evaluation time ratio and RMSE ratio, respectively. Each bar indicates the result for each dataset while the numbers after bagging indicates the sampling rate for each bootstrap. For the training time of bagging SVR, MDS resulted between “Bagging, 30%” and “Bagging, 50%”. The training time of “Bagging, 10%” was very fast. However, for the evaluation time, bagging took longer time than MDS. Since bagging employs all base learners to evaluate a test data point, the evaluation complexity of bagging is relatively high. In terms of RMSE, MDS was more accurate than bagging on average. For details, the experimental results are summarized by averaging over all datasets in Table 3.2. The accuracy of those bagging SVRs was lower than MDS. Bagging SVRs used less training time than MDS only when the sampling rate of each bootstrap of bagging was lower than 50% of the original data. On the other hand, the evaluation time of bagging SVRs

Table 3.3: Summary of the experimental results for data selection.

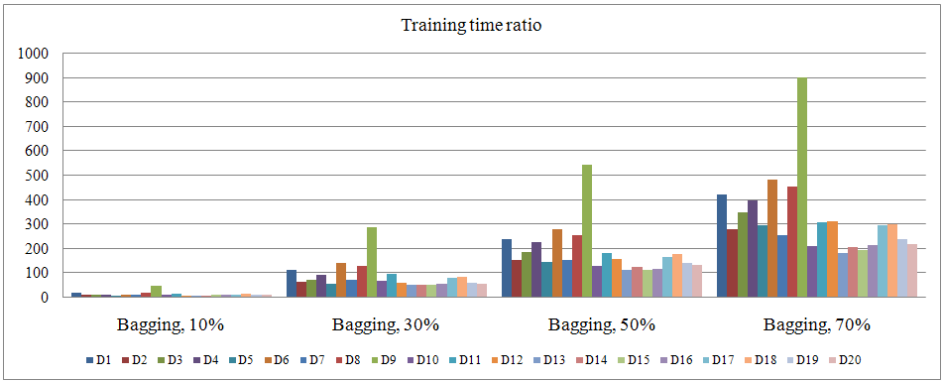
		VDS	MDS (α=0)	MDS (α=0.5)	MDS (α=1)
Fraction of RMSE to train all data (%)	Average	101.04	101.24	100.11	99.98
	Min	95.91	98.66	95.99	98.37
	Max	106.23	106.25	102.60	100.36
Fraction of training time to train all data (%)	Average	38.20	38.17	53.99	67.40
	Min	2.76	2.67	2.84	3.33
	Max	66.99	72.74	86.81	113.66
Sensitivity (%)	Average	86.01	85.84	92.79	96.27

resulted in 2∼8 times that of MDS.

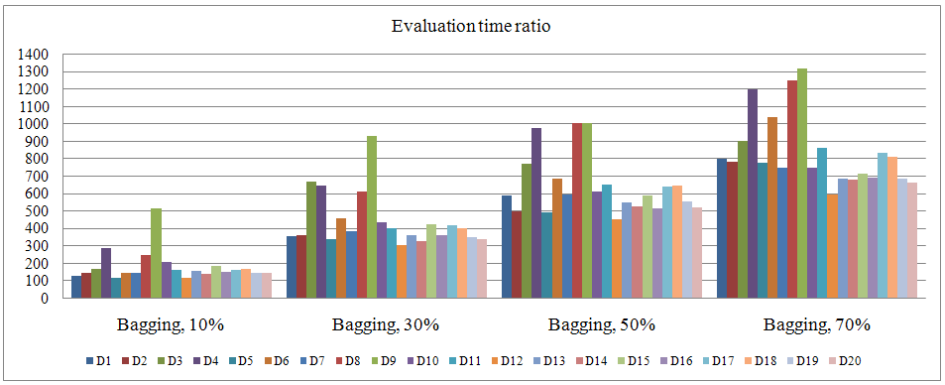
All experimental results were summarized in Table 3.3. Table 3.3 shows the fraction of RMSE of VDS and MDS compared to RMSE of the case of training all data in percentage and training time of VDS and MDS compared to that of the case of training all data in percentage. The training performance of VDS is very similar to MDS with $\alpha = 0$. However, the α increases, the performance of MDS overcome the performance of VDS. This experimental result showed that MDS is a extended and general version of VDS, as mentioned earlier. On average MDS with $\alpha = 0$ can train SVR using only 38.17 % of the original training time and a 1.24 % accuracy degradation. In case that requires higher model accuracy, the user can set a larger α. When $\alpha = 0.5$, the experimental results showed that MDS degraded RMSE by only 0.11 %, but the overall training time, including data selection time, took only half of the original training time. Moreover, MDS with $\alpha = 1$ trained SVR in 67.40 % of original training time, but that did not degrade the results. MDS can select on average of 85.84 %, 9.79 % and 96.27 % of actual support vectors when α was 0 , 0.5 and 1, respectively.

3.4 Parameter Analysis

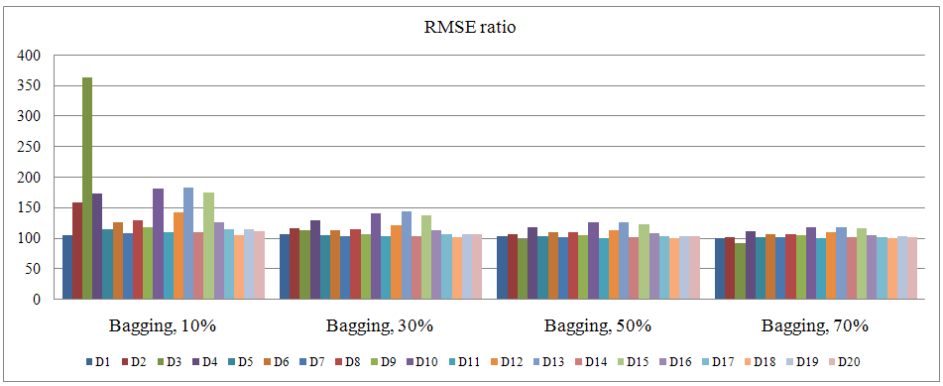
The MDS method requires two parameters for building multiple sample sets. l is the number of bootstraps while m % of n data are randomly included in a sample set. If l and m increases, more sample sets and more samples are involved. Larger l and m values guarantee the increased accuracy of MDS. If l and m become too large, however, then MDS takes too much time. Hence, it is important to select the most efficient and



(a)

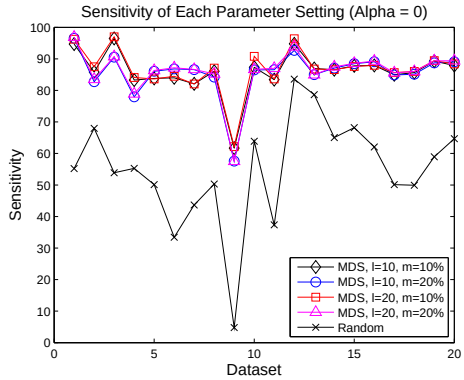


(b)

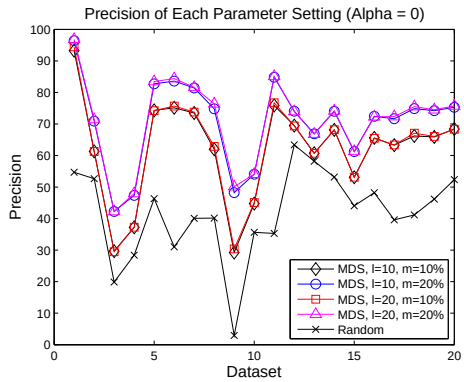


(c)

Figure 3.8: Comparison of MDS and bagging. (a) Training time ratio, (b) evaluation time ratio and (c) RMSE ratio.



(a)

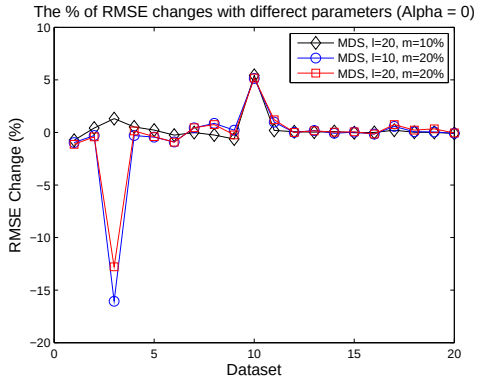


(b)

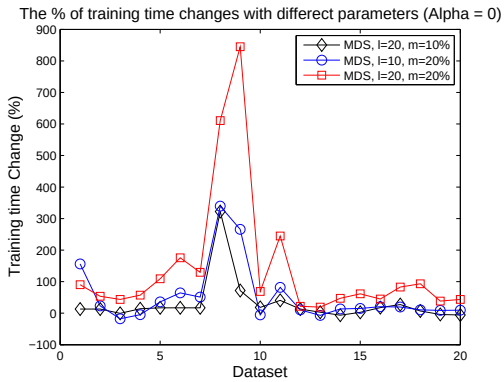
Figure 3.9: Sensitivity and precision of different parameter settings.

effective set of l and m. To observe the effects of values of different l and m, experiments were conducted with different parameter sets, $l \times m = \{10, 20\} \times \{10, 20\}$.

Since, MDS is designed to select support vectors, we must determine whether MDS accomplishes that task. Figure 3.9 showed the sensitivity (a) and the precision (b) of different parameter settings. The sensitivity and the precision can be calculated as in Eq. 3.5 and Eq. 3.6, respectively. For comparison, the sensitivity and the precision of the random sampling is also plotted. For all datasets and all parameter settings, the sensitivity and the precision of MDS is higher than random sampling. It seems that MDS tends to select more support vectors and less non–support vectors



(a)



(b)

Figure 3.10: The percentage of RMSE and the training time changes by different parameter settings.

and random sampling. The sensitivity of MDS is about 86%. That means MDS has a capability of selecting most of support vectors. Parameter l and m are not sensitive to select support vectors, but parameter m is slightly sensitive to avoid selecting non–support vectors.

$$\text{Sensitivity} = \frac{\text{The } \sharp \text{ of actual support vectors selected}}{\text{The } \sharp \text{ of actual support vectors}} \tag{3.5}$$

$$\text{Precision} = \frac{\text{The } \sharp \text{ of actual support vectors selected}}{\text{The } \sharp \text{ of data selected}} \tag{3.6}$$

Then the relation between parameter settings and model performances is compared. Figure 3.10 (a) shows the changes of RMSE with different parameters compared to the default parameter setting, $l = 10$ and $m = 10$, in terms of percentage, while Figure 3.10 (b) shows the changes of training time in the same manner. RMSE changes were approximately 0 % for 17 datasets among 20 datasets. However, the training time changes were dramatically large. Especially, the training time is very sensitive to m. When $m = 20$, the training time increased by more than 100 %, while the RMSE was almost the same. The conclusion is that even if l and m affect the precision, l and m rarely affect RMSE, the model accuracy. However, l and m affect the training time, dramatically. Hence, the default parameter setting, which was $l = 10$ and $m = 10$, is effective and efficient.

3.5 Summary

Research efforts have focused on reducing the training complexity of SVM. Since the training complexity of SVM is highly correlated to the number of training data, data selection is an effective method in order to reduce the training complexity of SVM. In addition, since data selection is working as a preprocessing step, data selection method can be applied with other training complexity reducing methods, such as decomposition method based SVM or a linear SVM. Moreover, the memory space can be saved by storing only a smaller subset of selected data, not all original data.

This Chapter proposed two data selection methods, VDS and MDS, to reduce the training time complexity of SVR using the characteristics of support vectors. Only those data that were likely to become support vectors were selected and used for training. VDS is a preliminary method using the binary voting method while MDS is an extended and general version of VDS. VDS selected data with binary voting of multiple sample learning. MDS estimated the averaged margin of training data, and those data which have the averaged margin greater than ε are selected to train. Both VDS and MDS automatically determined the number of data selected, which is a key factor in obtaining good results, and makes MDS less ambiguous. The parameter α controls the number of data selected by the model. This allows the model to adapt to the various noise level. With a high α, the sensitivity of selecting support vectors can be increased (see Figure 3.11). Through the experiments including 20 datasets,

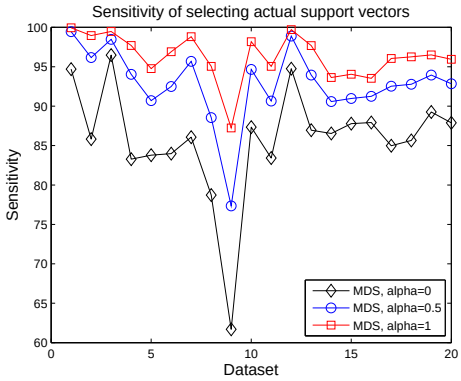


Figure 3.11: Sensitivity of MDS for each dataset.

MDS provided better generalization performances than the other benchmark methods. MDS performed well for diverse datasets: artificial, time series, non–time series, linear, non–linear, low noise or high noise.

Another strong point of the proposed methods are that it has fewer critical parameters than benchmark methods. For example, it is known as an NP–hard problem to find the optimal k of the k–NN based method. Moreover, several threshold parameters affect the results but are ambiguous to users. There is no guideline for those parameters, and a parameter set which is determined to be the best for a certain dataset is rarely the best for other datasets. However, MDS, only the parameters l and m, which do not affect the result directly, and α can be bounded between 0 and 1. Based on our analysis of the model parameters, the default parameter setting was very effective.

There are some limitations of the current work. First of all, the data redundancy can occur. MDS selects data which are likely to become support vectors by estimating margin of training data. Hence, MDS tends to select too many data if the number of support vectors are too large or the parameter ε is too small. Other methods, such as HSVM and k–NN based method, have focused on selecting a representative data point from a dense region in order to reduce redundant data. An additional effort that selecting representative data from a dense region to avoid redundancy can be employed to selected data by MDS. Second, the parameter analysis of MDS needs to be more detailed. MDS has two parameters: l, the number of multiple sample set and m, the number of data in a sample set. Though

the experiments, some guidelines for those parameters can be obtained and they were effective. However, the fundamental bounds for parameters are not researched. Efforts to determination of the fundamental bounds are future research area.

Chapter 4

Data Generation and Selection for Semi–Supervised Support Vector Regression

For SS–SVR, there are two common stages: (1) labeling the unlabeled data and (2) training models with the labeled data and the unlabeled data labeled from the first stage. In the first stage, the labels of the unlabeled data are estimated to use the unlabeled data in the construction of the final regression model. In the second stage, the final regression model is trained with the labeled and the unlabeled data in order to obtain a model which can be applied to unseen test data. The main motivation of SS–SVR is to adapt the great generalization performance of SVR by using SVR for the final model. For SS–SVR, the usage of the unlabeled data should be fit to train the final SVR model. In addition, the training complexity problem of SVR and SSL can be considered.

In this dissertation, the proposed SS–SVR algorithm can be summarized in four detail stages: (1) labeling the unlabeled data, (2) data generation, (3) data selection and (4) training the final SVR model. In the first stage, the unlabeled data are labeled using 2–PLR regression methods. Contrast to the conventional co–training method, the label of each unlabeled data point is estimated as the Gaussian probabilistic distribution form, not a single value. In addition, the iterative learning of co–training is not employed in order to reduce the training complexity of SS–SVR. In the second stage, the training data are generated from the unlabeled data

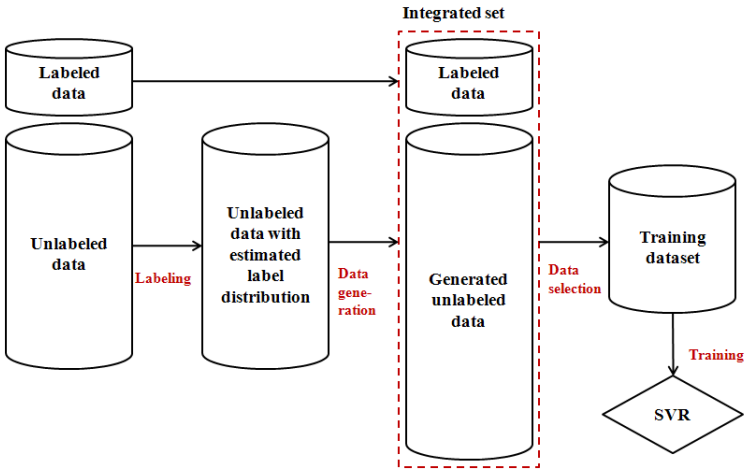


Figure 4.1: The overall procedure of the proposed SS–SVR method.

and their estimated label distribution. With the uncertainty of estimated labels for the unlabeled data, the data generation rate is varied. The unlabeled data with high uncertainty have high probability to be generated while the unlabeled data with low uncertainty have low probability to be generated. The integrated set is constructed by the labeled data and the generated unlabeled data. In the third stage, the data selection method is applied. Since the size of training dataset is increased by the data generation of the second stage, the data selection method should be employed in order to decrease the training complexity. In the final stage, the final SVR model is trained by the selected set from the third stage. The overall procedure of the proposed method is illustrated in Figure 4.1 while the notation is presented in Figure 4.2.

4.1 Labeling the Unlabeled Data

In Section 4.1, the process of labeling the unlabeled data is presented. Normally, the training dataset consists of the labeled dataset and the unlabeled dataset. Let us consider the labeled data $L = \{L_x, L_y\}$ where L_x is the input variables of the labeled data and L_y is the label. And the unlabeled data can be notated as U. Since the unlabeled data do not have their labels, U has only U_x.

As mentioned earlier, the local topology based method is better than

Notations

L: The labeled data

L_x: The input variables of the labeled data

L_y: The target variable of the labeled data

U: The unlabeled data

U_x: The input variables of the unlabeled data

$\hat{y}$: The estimated label of the unlabeled data

D_I: The integrated data set

U_G: The generated data

D_S: The selected data set

p_i: Data generation probability

Parameters

k_{local}: The number of nearest neighbors for PLR_{local}

k_{global}: The number of nearest neighbors for PLR_{global}

t: The number of trials for data generation

Figure 4.2: Notations for SS–SVR.

the function based method when labeling the unlabeled data. Because the local topology based method can describe the unlabeled data surroundings. On the other hand, the function based method for labeling the unlabeled data has a risk of an interpolation, which rarely gives new information of the unlabeled data. In order to obtain the estimated label distribution of the unlabeled data, PRL is employed. PLR is a key parameter to set, the number of nearest neighbors k. Even PLR was designed to be a k–invariant method, however, there still exists a risk for the noisy data when estimating the label distribution.

In order to overcome that risk and obtain the robust results, two PLR models are employed. The first model, PLR_{local}, plays a role in focusing on the local area of each unlabeled data point with a small k. This model captures the local topology of nearest labeled data for each unlabeled data. However, PLR_{local} can be sensitive to noisy data. The second model, PLR_{glocal}, plays a role in focusing on the global distribution of the labeled data with a large k. PLR_{glocal} may be worse to capture the local topology than the PLR_{local}. However, PLR_{glocal} can be good at capturing the global distribution of the labeled data. Hence, those two PLR models play each role with different k: k_{local} and k_{global}. Figure 4.3 (a) depicts

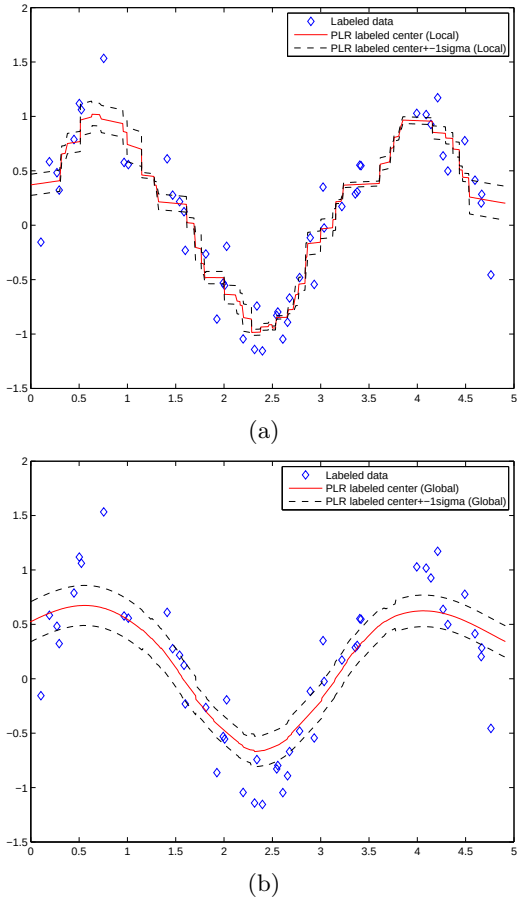


Figure 4.3: The output of PLR_{local} (a) and PLR_{global}.

the output of PLR_{local} focusing on the local topology while Figure 4.3 (b) depicts the output of PLR_{global} capturing the global distribution. PLR_{local} captured the local topology well, but the target variance was sensitive to noisy data. On the other hand, PLR_{global} captured the global distribution of the labeled distribution.

The outputs of both PLRs are obtained as a Gaussian probabilistic distribution form, $\hat{y}_i = N(\bar{y}_i, \sigma^2)$. To use the label distribution, not a label value, the final output combined the output of 2–PLR is needed to also a Gaussian probabilistic distribution form. Hence, the conjugation method which is widely used for Bayesian method (Bishop, 2006; Duda

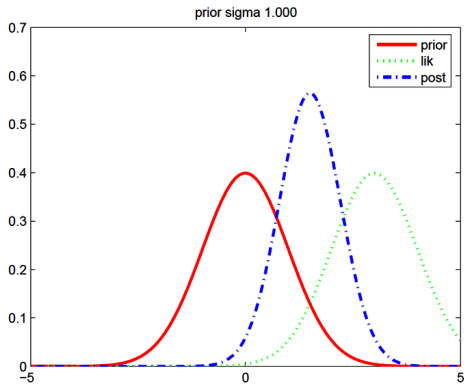


Figure 4.4: The conjugated new Gaussian distribution.

et al., 2001; Mitchell, 1997) was employed. The prior probability and the likelihood probability can be conjugated as the posterior probability as shown in Figure 4.4. Since PLR$_{local}$ has more near data–driven view, it mimics the likelihood of the Bayesian method. On the other hand, since PLR$_{global}$ focuses on the global distribution of the underlying function, it mimics the prior of the Bayesian method. The conjugated $\bar{y}$ and the conjugated σ^2 can be calculated in Eq. 4.1 and Eq. 4.2, respectively.

$$\bar{y}_{conjugate} = \frac{\frac{\bar{y}_{global}}{\sigma^2_{global}} + \frac{n \times \bar{y}_{local}}{\sigma^2_{local}}}{\frac{1}{\sigma^2_{global}} + \frac{n}{\sigma^2_{local}}}. \tag{4.1}$$

$$\sigma^2_{conjugate} = \frac{1}{\frac{1}{\sigma^2_{global}} + \frac{n}{\sigma^2_{local}}}. \tag{4.2}$$

4.2 Data Generation

The conventional SSL method uses the single label value, $\bar{y}$. However, some drawbacks exist. One drawback is that the uncertainty of labeling the unlabeled data is not considered. Some unlabeled data located in a dense region constituted by the labeled data are well–labeled by the local topology model. On the other hand, the estimated labels of some other unlabeled data located in a sparse region have much uncertainty. If the uncertainty is not considered, the final regression method may train wrong labels of the unlabeled data. Another drawback occurs when training

SVR. SVR employs the ε–tube and maximizes margins of the training data. Hence, the training data should be distributed in a margin region of the regression function. Machine learning–based regression methods estimate the labels based on interpolation. As shown in Figure 4.5, the single labels are formed narrowly around the underlying function. Then, maximum margin model may give an arbitrary results. In addition, the single labels are located in the labeled data surroundings because of the interpolation. The unlabeled data do not have any additional information of the underlying function, but just give redundant information of the training data. Then, only a few labeled data is selected to be support vectors.

In order to overcome the drawbacks of the conventional SSL method, the data generation was conducted. Contrast to the conventional SSL method, the proposed method obtains the estimated label distribution of each unlabeled data point by 2–PLR and their conjugation. Data generation of the proposed method refers to a multiple random generation of training data from the unlabeled data, $U_{\mathbf{x}_i}$ and its estimated label distribution, $\hat{y}_i = N(\bar{y}_i, \sigma_i^2)$. Since $U_{\mathbf{x}_i}$ is a static term, a generated data point is described as $\{U_{\mathbf{x}_i}, \hat{y}_i = N(\bar{y}_i, \sigma_i^2)\}$. The integrated dataset, D_I, is constructed with the generated dataset of the unlabeled data point, U_G, and the labeled data: $D_I = L \bigcup U_G$. Data generation gives advantages for the SSL. First of all, data generation overcomes the problem occurred by the interpolation of the labeled data. Since the labels of the unlabeled data are generated from Gaussian distributions, the training data are located around the target function with some margins. Second, the uncertainty of labeling the unlabeled data can be considered. In PLR learning, σ_i^2 is calculated related to the variation of the nearest neighbors of each data point, $\mathbf{x}_i$. Hence, the larger σ^2, the sparser the nearest neighbors is located around the unlabeled data point. Hence, those unlabeled data which have large σ_i^2 can be considered to have high labeling uncertainties. On the other hand, those unlabeled data which have small σ_i^2 can be considered to have low labeling uncertainties.

In data generation stage, each unlabeled data point, $U_{\mathbf{x}_i}$, has different generation rate. For the unlabeled data with high uncertainties, more training data should be generated in order to generate more information of that region. Moreover, if σ_i^2 is large, which means the confidence of $\bar{y}_i{}^2$ is too low, the single $\bar{y}_i{}^2$ has low probability to be the labels of $U_{\mathbf{x}_i}$. Hence, for those unlabeled data, the probability to be generated should be

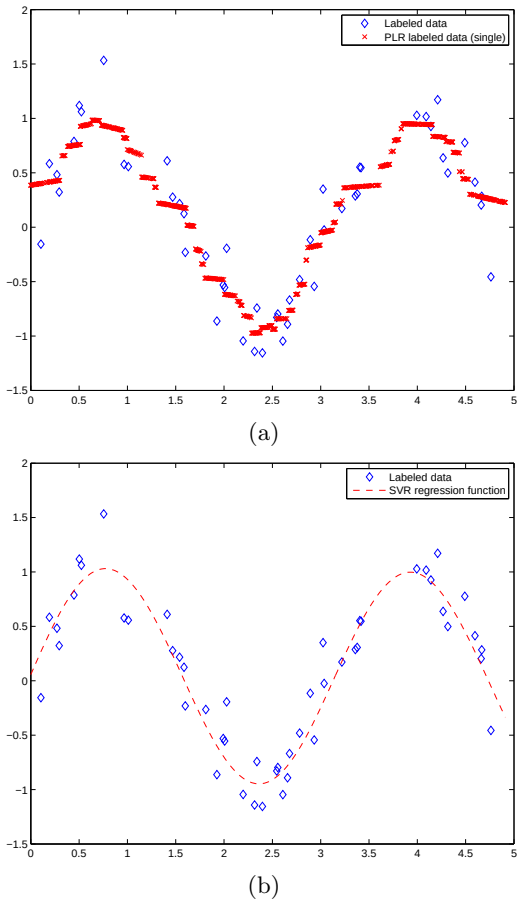
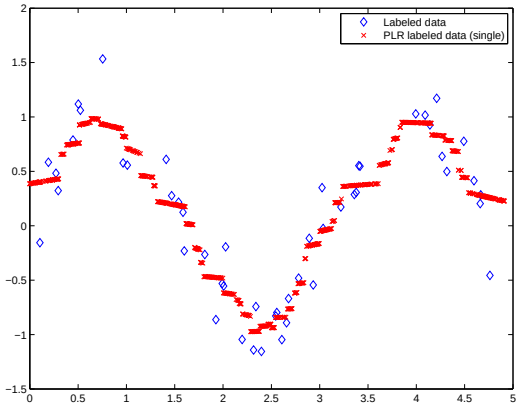
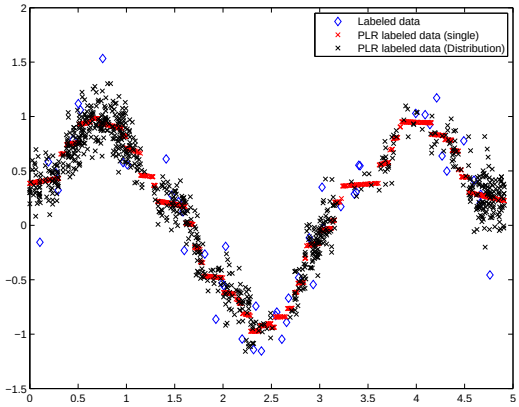


Figure 4.5: The regression output from PLR (a) and SVR (b).

high. On the other hand, for the unlabeled data with low uncertainties, less training data should be generated in order to prevent the duplication. Similar training data can be generated with small σ_i^2. Those training data increase the training complexity, but do not give additional information we have expected to the unlabeled data. Hence, the generation rate should be proportional to σ_i^2. In this research, the data generation probability, p_i is calculated by scaling of σ_i^2 as in Eq. 4.3. Every unlabeled data have t numbers of independent trials to be generated. p_i determines whether an unlabeled data point is generated or not for every t trials, independently. Then, those unlabeled data with high uncertainty have more chances to



(a)



(b)

Figure 4.6: Integrated dataset constructed with the unlabeled data ((a) single label values and (b) generated data).

be generated while others have less chances. Figure 4.6 compares the integrated dataset with the labeled data and the unlabeled data with their single labels (a) and the integrated dataset constructed by the labeled data and the generated data (b). The data generation gives more information of the unlabeled data located in an uncertain region.

$$p_i = \frac{\sigma_i^2 - min(\sigma^2)}{max(\sigma^2) - min(\sigma^2)}. \tag{4.3}$$

4.3 Data Selection and Support Vector Regression

SSL is a time consuming method. The basic assumption of the SSL is that the number of the unlabeled data are very large, 10∼100 times of the number of the labeled data. Moreover, the proposed method employs the data generation step. The size of training dataset is even larger than the original training dataset. Hence, an additional method is needed to decrease the training complexity. Since the proposed method is designed for SS–SVR, the training time complexity follows that of SVR. The main issue of the training complexity of SVM is the number of training data. The training complexity of SVR is strongly correlated to the number of training data: $O(n^3)$ of the training time complexity and $O(n^2)$ of the training memory complexity, where n is the number of training data. Hence, the data selection method proposed in Chapter 3, MDS, is employed to reduce the training complexity of SVR. The algorithm of the proposed method is summarized in Figure 4.7.

4.4 Experimental Results

4.4.1 Experiment Setting

For the experiments, 18 benchmark datasets, including one artificial dataset and 17 real–world datasets, were used. Real–world benchmark datasets were gathered from Delve datasets[1], Time Series Data Library (TSDL)[2] and Statlib[3]. All datasets are summarized in Table 4.1.

[1]Delve Dataset: `http://www.cs.toronto.edu/~delve/data/datasets.html/`

[2]TSDL: `http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/`

[3]Statlib: `http://lib.stat.cmu.edu/datasets/`

The Proposed SS–SVR Method Algorithm

1. Initialization

k_{local}: The number of nearest neighbors for PLR_{local}

k_{global}: The number of nearest neighbors for PLR_{global}

t: The number of trials for data generation

ADD_{U_x}: An empty set for the unlabeled input data

ADD_{U_y}: An empty set for the estimated labels

2. Labeling the unlabeled data

$N_{local} \leftarrow \text{PLR}_{local}(L_x, L_y, U_x, k_{local})$

$N_{global} \leftarrow \text{PLR}_{global}(L_x, L_y, U_x, k_{global})$

$N_{conjugate} \leftarrow \text{Conjugation}(N_{local}, N_{global})$

3. Data generation

$p_i = \frac{\sigma_i^2 - min(\sigma)}{max(\sigma) - min(\sigma)}$

FOR all $U_{\mathbf{x}_i}$

  FOR 1 to t

    $r \leftarrow Uniform(0, 1)$

    IF $p_i > r$

      $ADD_{U_x} \leftarrow ADD_{U_x} \bigcup U_{\mathbf{x}_i}$

      $ADD_{U_y} \leftarrow ADD_{U_y} \bigcup \hat{y}_i \sim N(\bar{y}_i, \sigma_i^2)$

    END IF

  END FOR

END FOR

$D_{I_x} = L_x \bigcup ADD_{U_x}$

$D_{I_y} = L_y \bigcup ADD_{U_y}$

4. Data selection

$\{Select_x, Select_y\} = \text{MDS}(D_{I_x}, D_{I_y})$

5. Run SVR

$\hat{y} = f(x) \leftarrow \text{SVR}(Select_x, Select_y)$

Figure 4.7: The algorithm of the proposed SS–SVR method.

Table 4.1: Datasets used in the experiments for SS–SVR

No.	Name	♯ Train	♯ Test	♯ Attribute	Origin	Feature
1	Add10	2000	2000	5	Delve Datasets	Art.
2	Santa Fe A	890	100	10	Santa Fe Comp.	T.S.
3	Santa Fe D	2000	2000	10	Santa Fe Comp.	T.S.
4	Santa Fe E	1490	500	10	Santa Fe Comp.	T.S.
5	Sun Spot	2000	1000	10	TSDL	T.S.
6	Melbourne Temp.	2000	1000	10	TSDL	T.S.
7	Gold	700	300	10	TSDL	T.S.
8	Wind	2000	2000	11	Statlib	Non–T.S.
9	Abalone	2000	2000	10	Delve Datasets	Non–T.S.
10	Computer Activity	2000	2000	12	Delve Datasets	Non–T.S.
11	Bank 8FM	2000	2000	8	Delve Datasets	Non–T.S.
12	Bank 8NH	2000	2000	8	Delve Datasets	Non–T.S.
13	Pumadyn 8FM	2000	2000	8	Delve Datasets	Non–T.S.
14	Pumadyn 8NH	2000	2000	8	Delve Datasets	Non–T.S.
15	Census House 8L	2000	2000	8	Delve Datasets	Non–T.S.
16	Census House 8H	2000	2000	8	Delve Datasets	Non–T.S.
17	Census House 16L	2000	2000	16	Delve Datasets	Non–T.S.
18	Census House 16H	2000	2000	16	Delve Datasets	Non–T.S.

The features of regression datasets can be partitioned into three types. “Art.”, “T.S.” and “Non–T.S.” indicates an artificial dataset, a time series dataset and a non–time series multi–variate dataset, respectively. Add10 dataset is another artificial dataset gathered from the Delve datasets. We used only five relevant input features, excluding five noise terms. Time series datasets were reformulated as regression problems by using the previous 10 values to estimate the following single value, which is a typical way to solve time series problems. The Wind dataset was reformulated to estimate the wind speed of the Dublin station using other 11 observed stations’ wind speeds. The dataset 9 to the dataset 18 from Table 4.1 came from three datasets: Bank, Pumadyn and Census House. The number following the dataset name denotes the number of features used. FM, NH, L and H stand for ‘fairly linear–moderate noise’, ‘nonlinear–high noise’, ‘low task difficulty’ and ‘high task difficulty’, respectively. Since one dimensional dataset is not suitable for SSL problems, the one dimensional dataset used in Chapter 3 was not analyzed. We expected to analyze the model performances varied with the different characteristics of training datasets, such as linearity and training complexity. To evaluate the performances, the original dataset was randomly split into training data and test data. The hyper–parameters of SVR were determined by cross–validation with $\mathrm{C} \times \varepsilon = \{0.1, 0.5, 1, 3, 5, 7, 10, 20, 50, 100\} \times \{0.01, 0.05, 0.07, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9, 1\}$. RBF kernel was used as a

kernel function and the kernel parameter σ was fixed to 1.0 for all datasets. All datasets were normalized.

The proposed method was compared to COREG$_{kNN}$, COREG$_{SVR}$ and Co–SVR. COREG is from Zhou and Li (2007). Both COREG–based methods employ k–NN for labeling the unlabeled data. The difference between COREG$_{kNN}$ and COREG$_{SVR}$ is that COREG$_{kNN}$ and COREG$_{SVR}$ employ k–NN and SVR for the final training part, respectively. The number of k was fixed to 3 and 5 for two k–NN base models used for COREG. Co–SVR is from Wang et al. (2010). The hyper–parameters for Co–SVR was set to the same values which were set to the original training dataset. The GA part for Co–SVR is omitted in order to reduce the training complexity. For the initialization of both co–training based methods, the labeled data were partitioned into two different training sets, randomly. The number of data in a working set of the unlabeled data is set to 100 and 40 for COREG and Co–SVR, respectively. The number of maximum iteration is set to 100 for all co–training based methods. The proposed method also has parameters to set. The number of nearest neighbors for 2–PLR were set to 5 and 20 for k_{local} and k_{global}, respectively. The number of trails for each unlabeled data in data generation, t, was set to 3, 5 and 7.

The original training data were randomly sampled to be the labeled data, and the rest of them were used as the unlabeled data. The labels of the unlabeled data were masked and not used in the training and the test. The ratio of the labeled data were 20%, 10%, 5% and 1% of the original data. Since the experimental results can be biased by samplings for the labeled section and the data generation, the experimental results were averaged over 10-time repetitions. The performances of each method are measured by RMSE (Eq. 3.4) and training time (sec.), including the SSL part.

4.4.2 Experimental Results

Figure 4.8-4.19 presents the experimental results for 18 datasets, from the labeled ratio is 20% to 1%. The pairs of RMSE and training time in seconds are plotted corresponding to each method. The closer a result is plotted to the origin, the better the method performs. The experimental results of COREG$_{kNN}$, COREG$_{SVR}$ and Co–SVR are plotted as upper triangles, lower triangles and diamonds, respectively. The experimental results of the proposed method with various t are plotted as squares connected with a line. The accuracy of the proposed method is

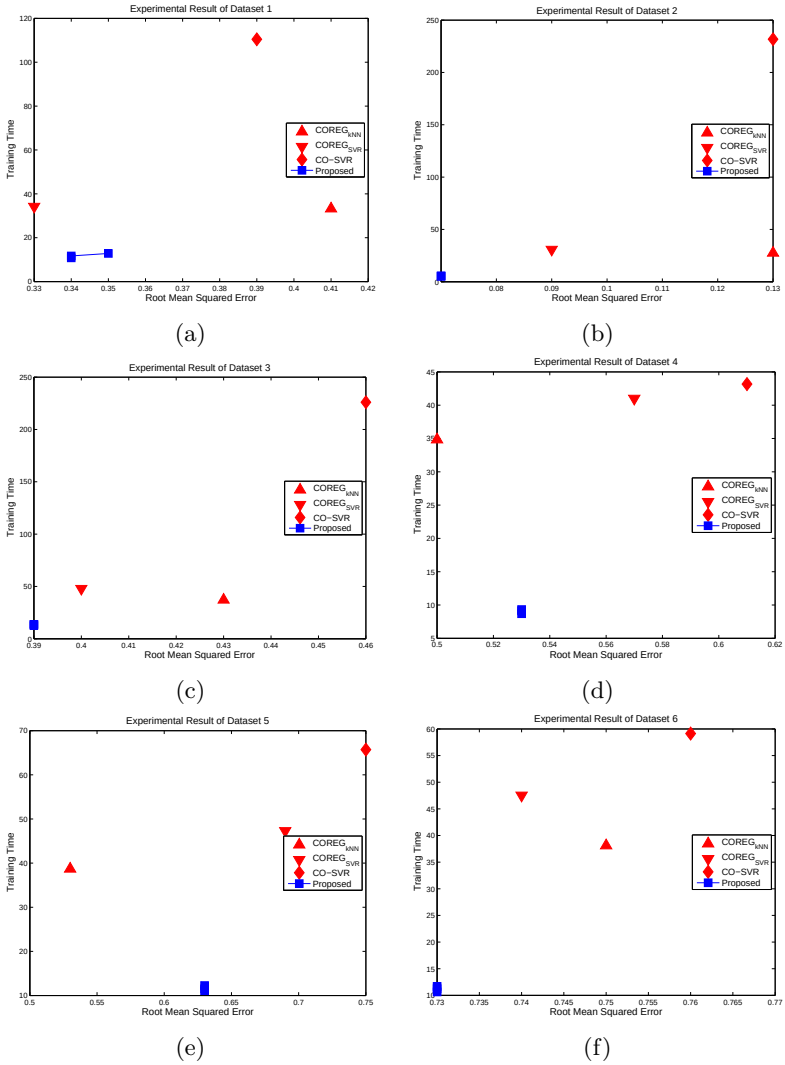
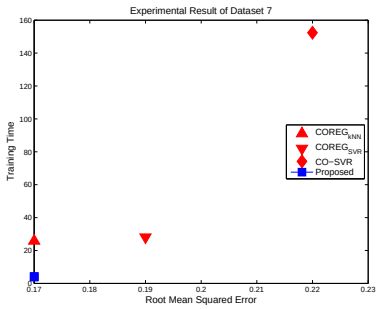
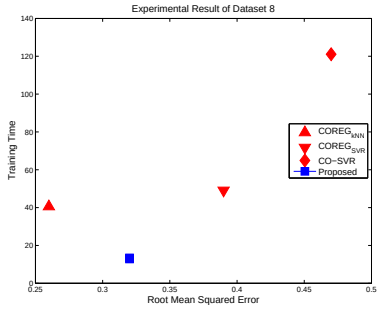


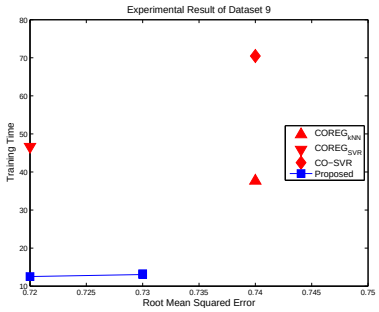
Figure 4.8: Experimental results when L=20% for D1 to D6.



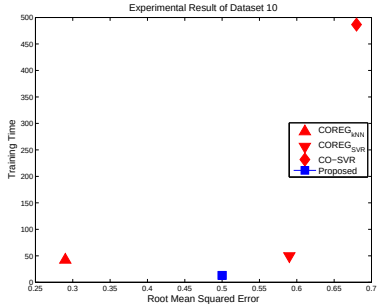
(g)



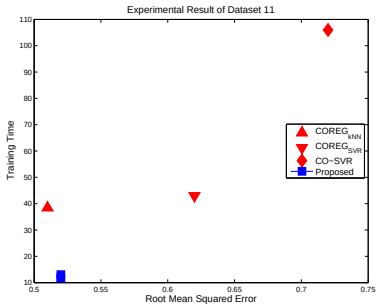
(h)



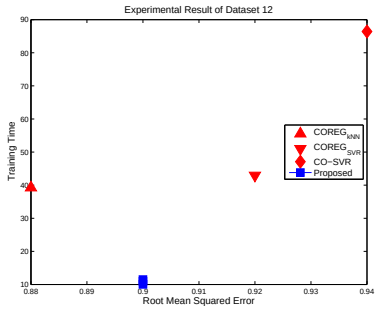
(i)



(j)

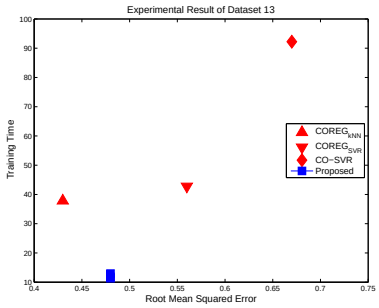


(k)

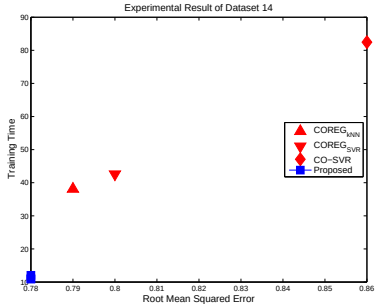


(l)

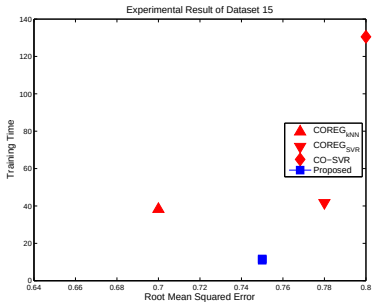
Figure 4.9: Experimental results when L=20% for D7 to D12.



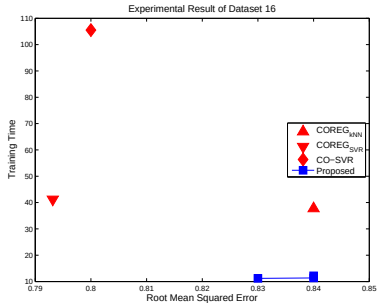
(m)



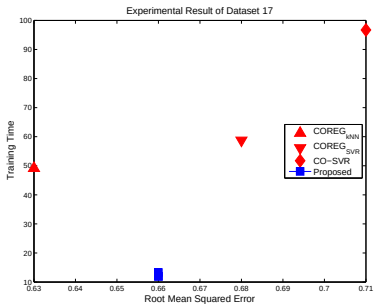
(n)



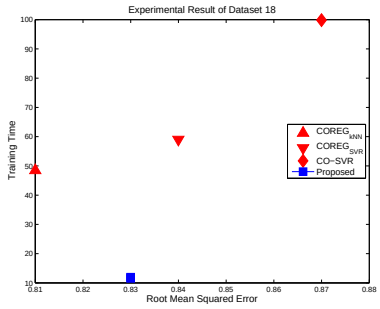
(o)



(p)



(q)



(r)

Figure 4.10: Experimental results when L=20% for D13 to D18.

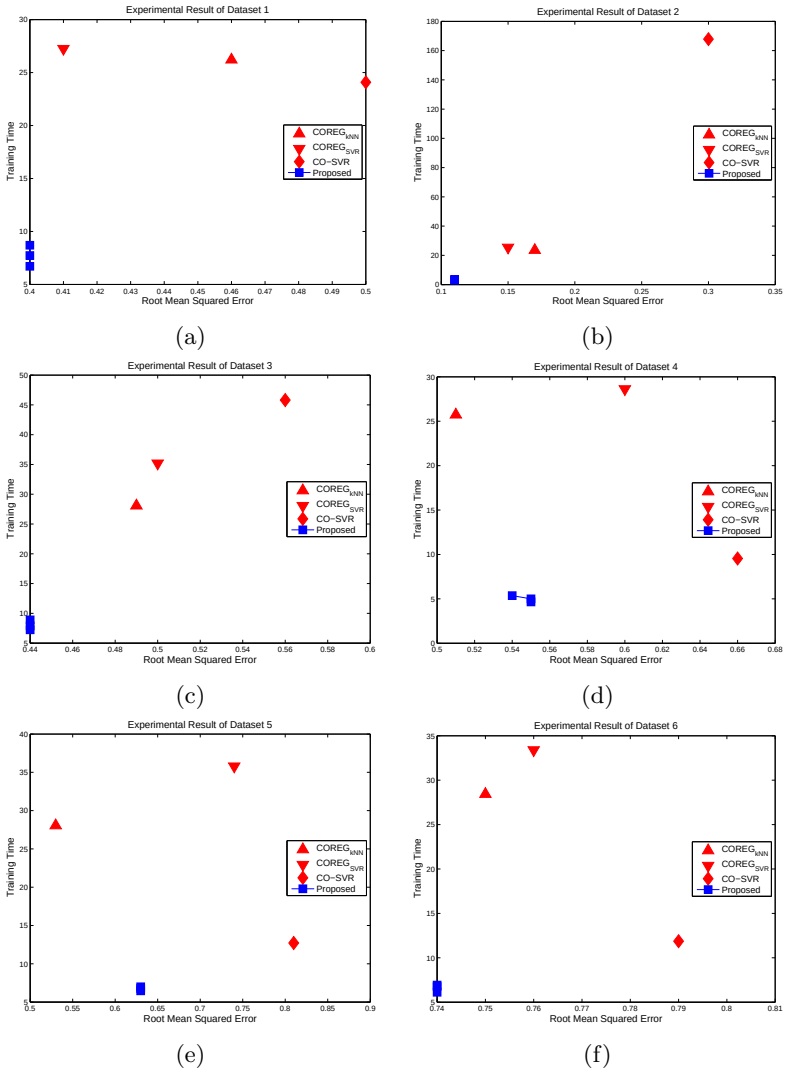
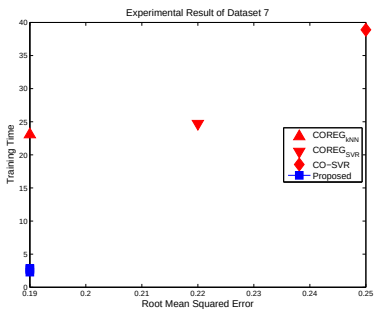
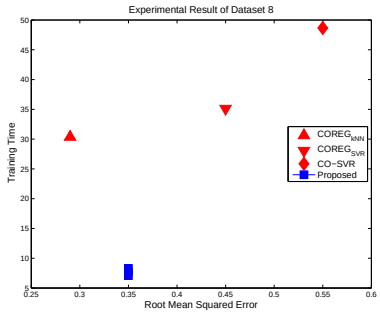


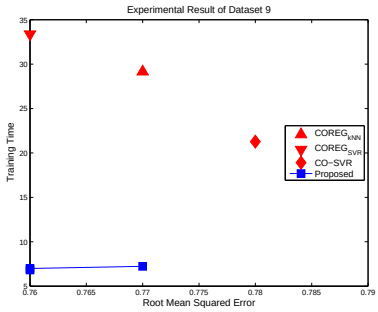
Figure 4.11: Experimental results when L=10% for D1 to D6.



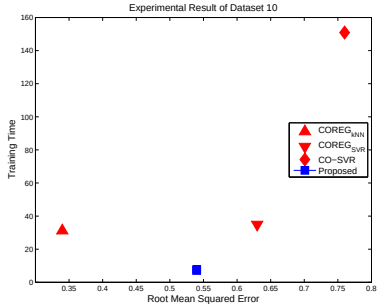
(g)



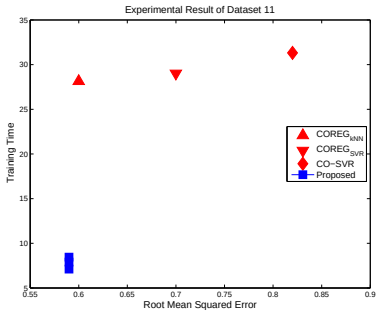
(h)



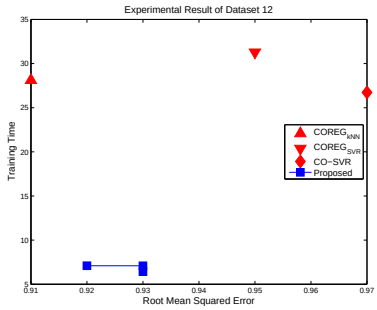
(i)



(j)



(k)



(l)

Figure 4.12: Experimental results when L=10% for D7 to D12.

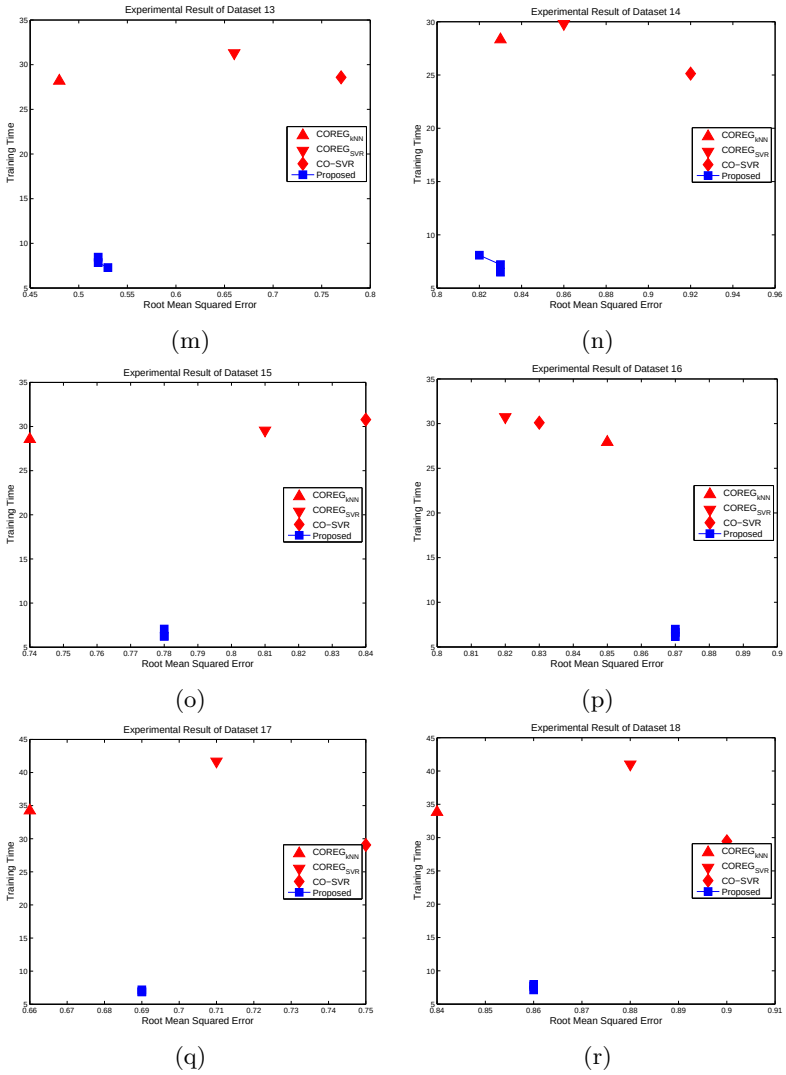


Figure 4.13: Experimental results when L=10% for D13 to D18.

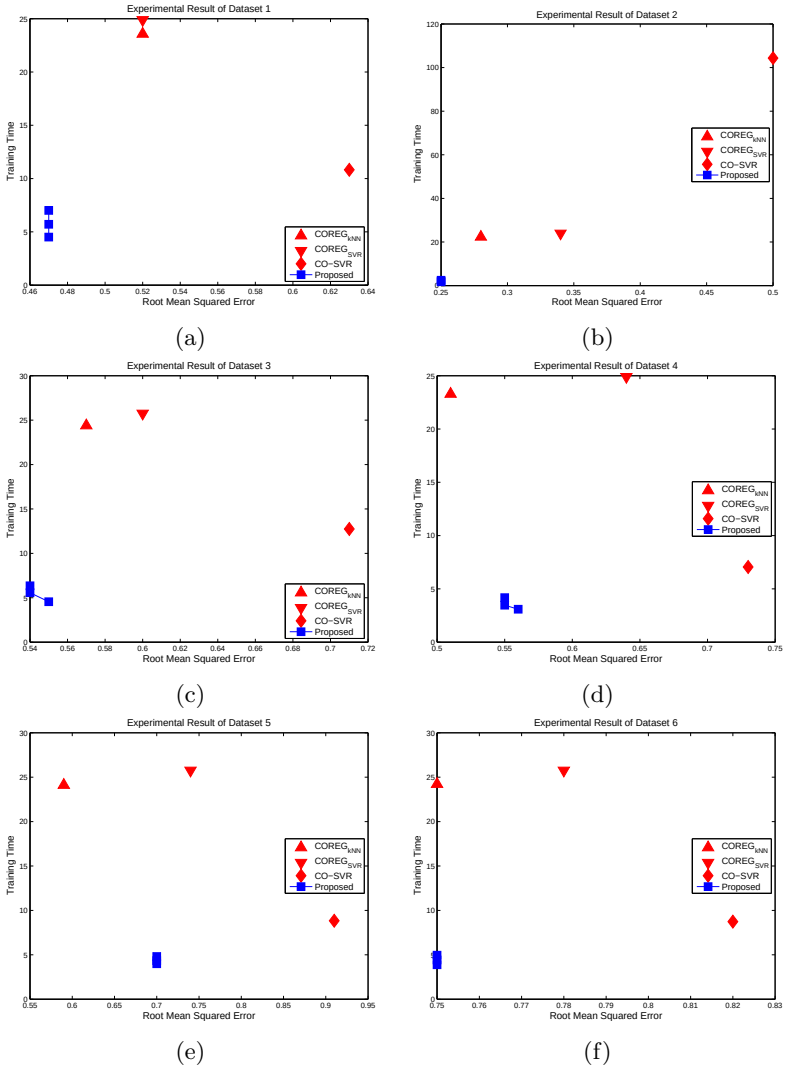
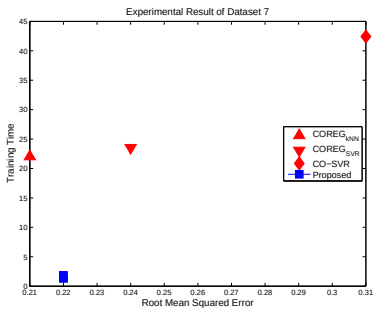
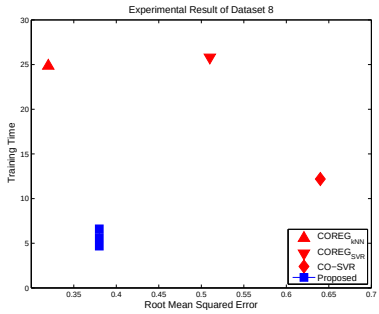


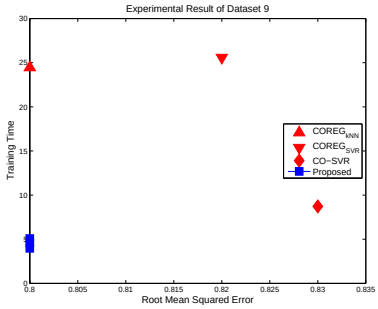
Figure 4.14: Experimental results when L=5% for D1 to D6.



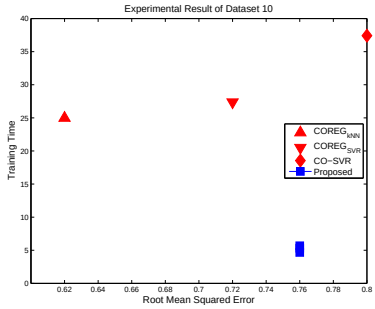
(g)



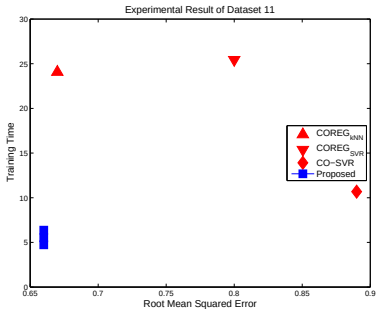
(h)



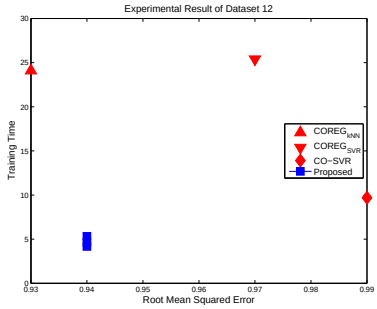
(i)



(j)

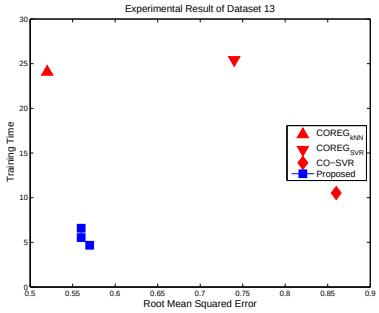


(k)

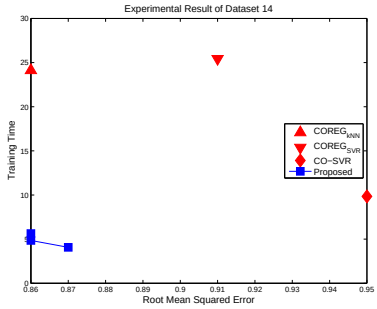


(l)

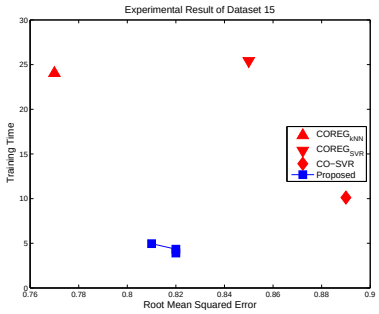
Figure 4.15: Experimental results when L=5% for D7 to D12.



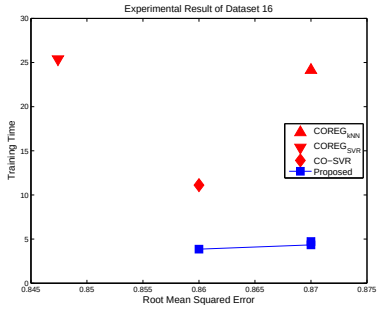
(m)



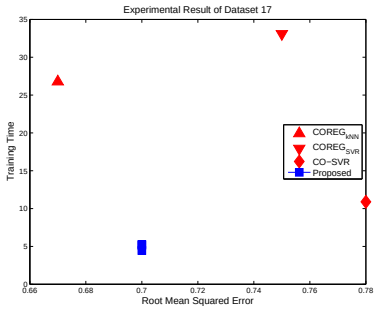
(n)



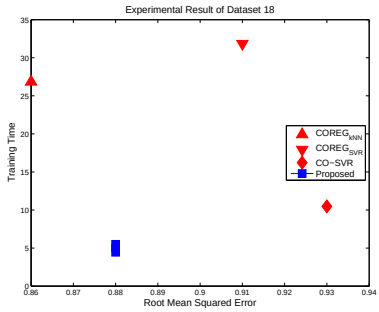
(o)



(p)



(q)



(r)

Figure 4.16: Experimental results when L=5% for D13 to D18.

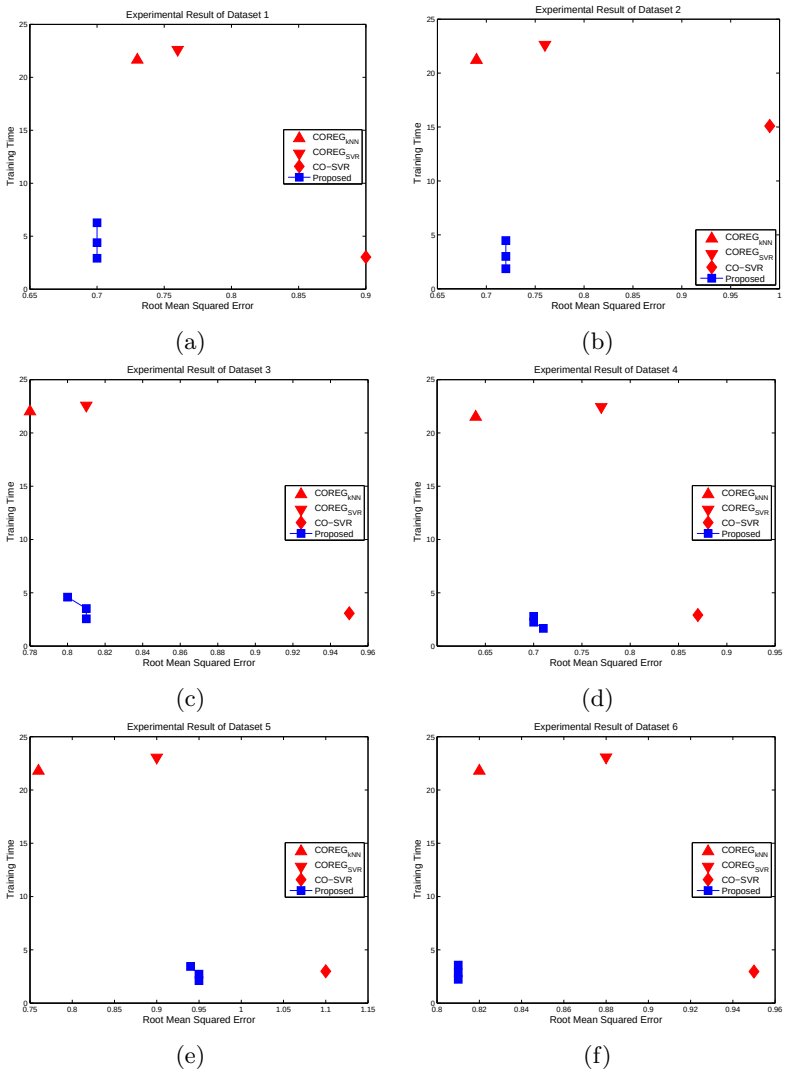


Figure 4.17: Experimental results when L=1% for D1 to D6.

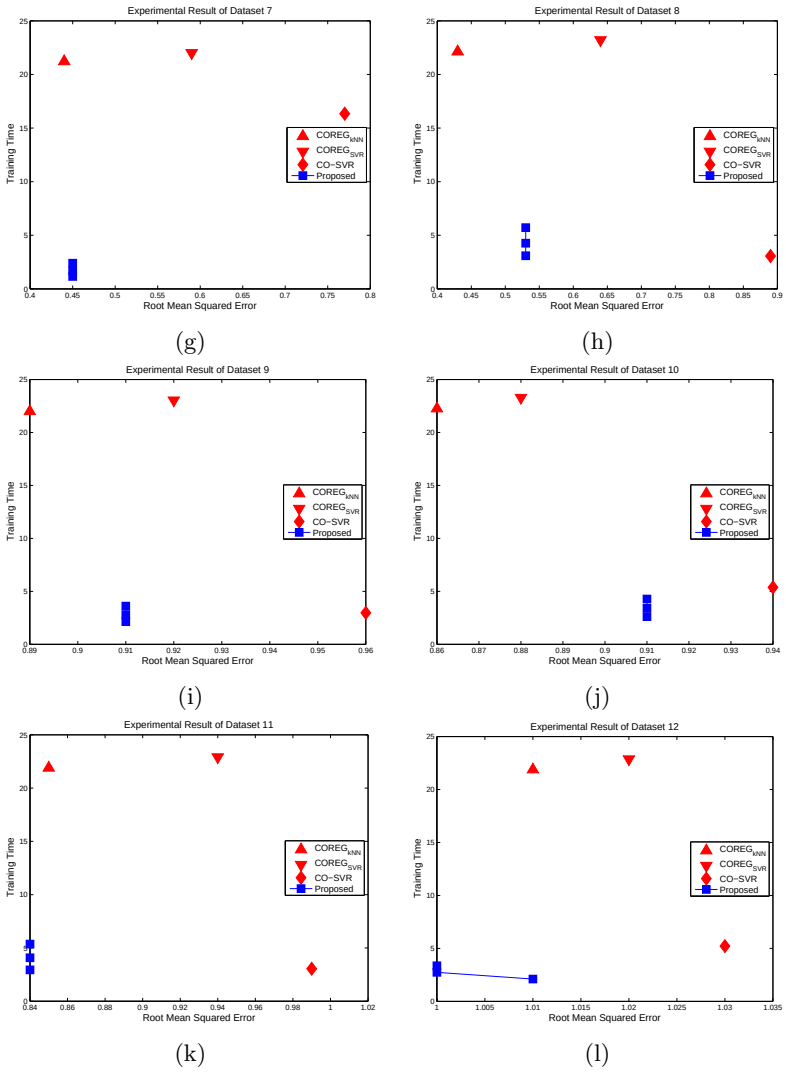


Figure 4.18: Experimental results when L=1% for D7 to D12.

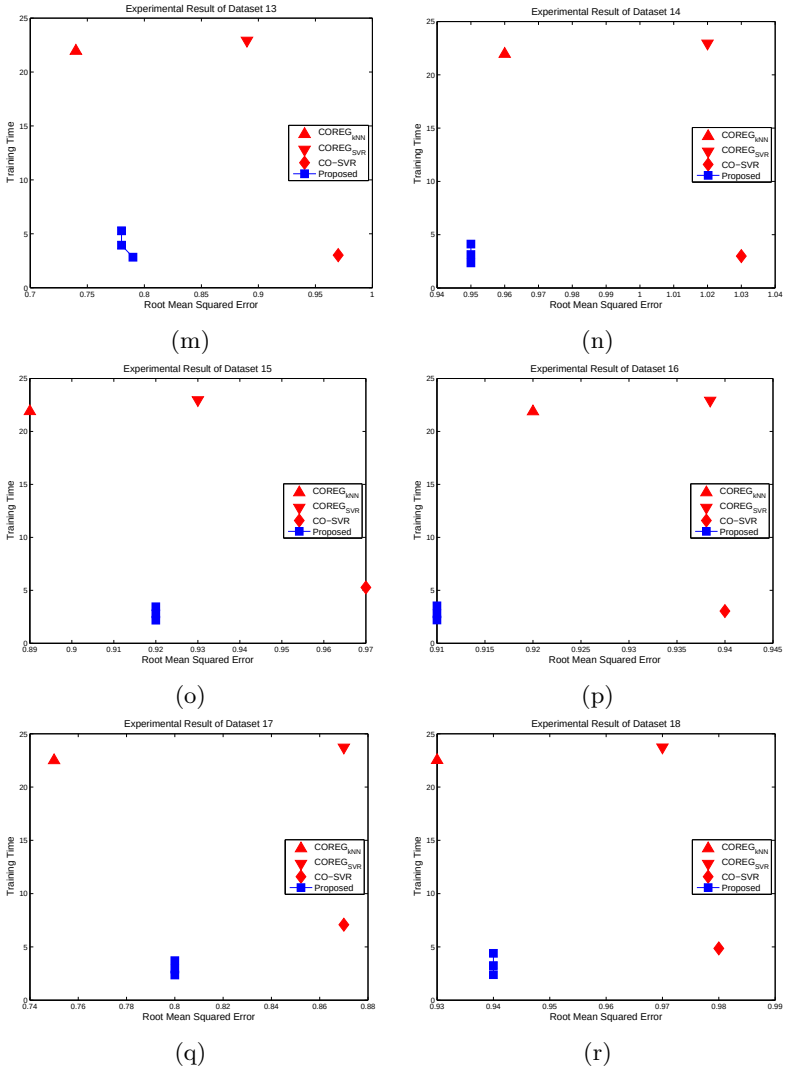


Figure 4.19: Experimental results when L=1% for D13 to D18.

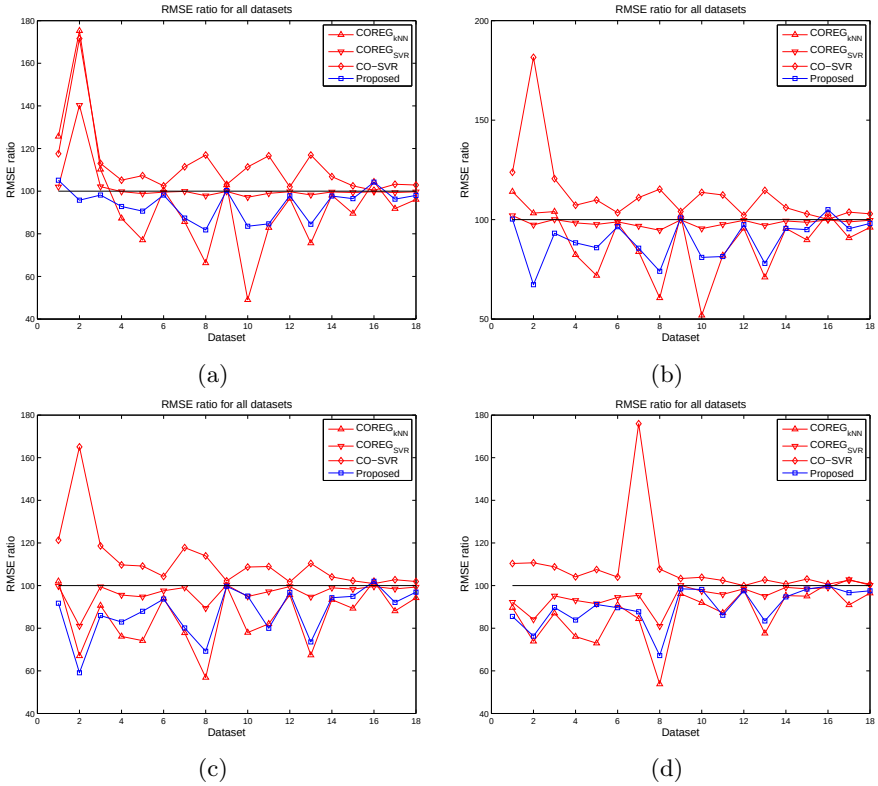


Figure 4.20: RMSE ratio for all datasets, (a) L=20%, (b) L=10%, (c) L=5% and L=1%.

comparable to COREG$_{kNN}$. However, the training time of the proposed method is about 30% of COREG$_{kNN}$. The proposed method showed better efficiency than COREG$_{kNN}$. On the other hand, among the proposed method, COREG$_{SVR}$ and Co–SVR, which employ SVR as the final regression method, the proposed method showed the best experimental results. The accuracy is smallest and the training time is shortest for all datasets. For the proposed method, the parameter t is not sensitive to most datasets. It seems that even a small number of trials, data generation can cover an effective range of training space.

Figure 4.20 illustrates RMSE ratio for all datasets while Figure 4.21 illustrates training time for all datasets. RMSE ratio is the fraction of RMSE of a method to RMSE of only labeled data trained. For exam-

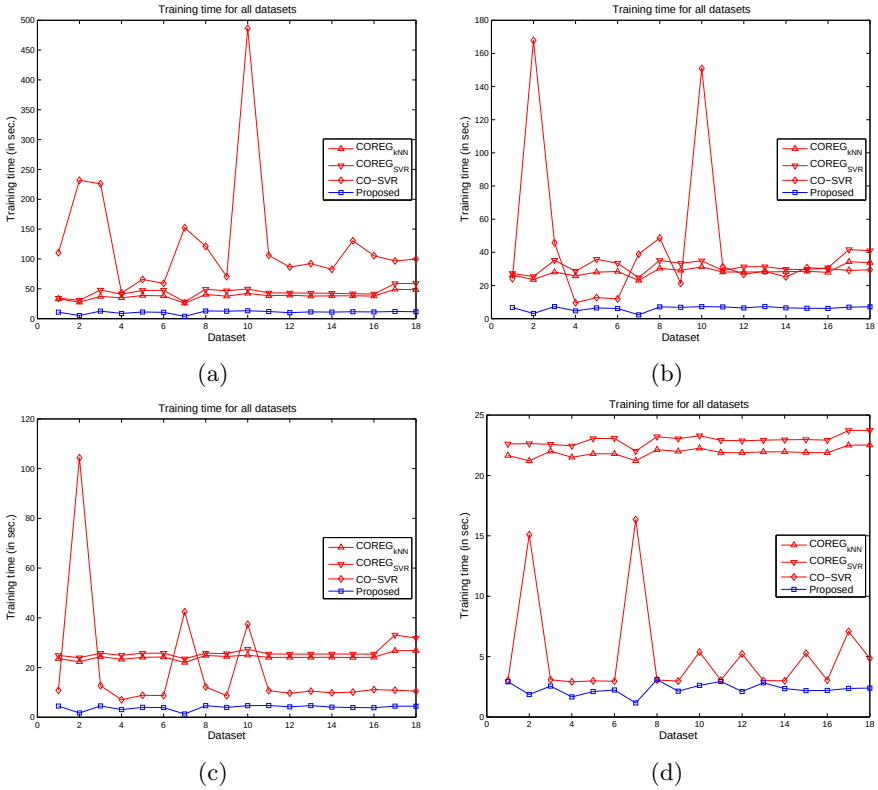


Figure 4.21: Training time for all datasets, (a) L=20%, (b) L=10%, (c) L=5% and L=1%.

Table 4.2: Summary of the experimental results for SS–SVR (RMSE ratio).

	L=20%	L=10%	L=5%	L=1%	**Avg.**
COREG$_{kNN}$	95.28	88.67	84.92	86.56	**88.86**
COREG$_{SVR}$	101.78	98.41	96.53	95.19	**97.98**
Co–SVR	111.74	113.07	111.32	108.28	**111.10**
Proposed, 1	95.47	91.26	88.79	91.60	**91.78**
Proposed, 3	94.08	89.92	87.55	90.09	**90.41**
Proposed, 5	94.16	89.99	87.52	89.86	**90.38**
Proposed, 7	94.21	90.01	87.48	89.72	**90.36**

Table 4.3: Summary of the experimental results for SS–SVR (training time).

	L=20%	L=10%	L=5%	L=1%	**Avg.**
COREG$_{kNN}$	37.95	28.40	24.24	21.89	**28.12**
COREG$_{SVR}$	44.12	32.12	26.15	22.94	**31.33**
Co–SVR	133.46	42.37	18.70	5.13	**49.42**
Proposed, 1	1.34	1.29	1.27	1.17	**1.27**
Proposed, 3	10.65	6.19	3.93	2.32	**5.77**
Proposed, 5	11.08	6.63	4.56	3.14	**6.35**
Proposed, 7	11.44	7.10	5.19	4.13	**6.97**

ple RMSE ratio of the proposed method is calculated as $RMSERatio = \frac{RMSE_{proposed}}{RMSE_{labeled}} \times 100$. The smaller RMSE ratio is the better improvement a method was. RMSE ratio measures the improvement of SSL than the conventional supervised learning. The RMSE ratio of the proposed method is stable regardless of the labeled ratio and datasets. The proposed method outperforms COREG$_{SVR}$ and Co–SVR, and is comparable to COREG$_{kNN}$. The training time of the proposed method showed the greatest among other benchmark methods.

The overall experimental results are summarized in Table 4.2 and Table 4.3. The number after "Proposed" indicates the parameter t. "Proposed, 1" is the experimental results of the proposed method without data generation. As shown in Table 4.2, the data generation stage can improve the accuracy than one without it. The proposed method outperforms all the methods employing SVR as a final model, COREG$_{SVR}$ and Co–SVR. The accuracy of the proposed method was about 1.5% higher than COREG$_{kNN}$. However, as shown in Table 4.3, the training time of the proposed method was only 20∼25% of training time of COREG$_{kNN}$. the proposed method can be concluded as a very efficient SS–SVR method.

The accuracy was stable when the parameter t has changed excluding $t = 1$ while the training time was differed. For these benchmark datasets, $t = 3$ was enough to train. The training time of "Proposed, 1" was very short. The reason was, without data generation, the estimated labeled of the unlabeled data were just interpolation of the labeled data. Hence, only a few number of support vectors from the labeled data is involved to be trained by SVR while the unlabeled data are just located inside the ε–tube.

4.5 Summary

In Chapter 4, a new method for SS–SVR was proposed. In SSL, a large number of the unlabeled data are used in training. Since the labels of regression data are continuous numbers, SSL regression method is more complex when estimating the labels of the unlabeled data than SSL classification method. Co–training, which employs regression models to estimate the labels of the unlabeled data, is the most popular approaches for SSL regression. However, the conventional co–training based SSL regression methods have some drawbacks. One big obstacle is the training complexity occurring because co–training approach employs iterative learning. And uncertainty of estimated labels of the unlabeled data tend to degrade the performance of SSL regression. Some co–training based SSL regression methods tend to conclude a self–training result because of the interpolation problem.

The proposed SS–SVR method employed the data generation to overcome the uncertainty issue, and employed the data selection to overcome the training complexity problem. In order to obtain the estimated label distribution for the unlabeled data, PLR, which is a probabilistic reconstruction method, was employed. Since PLR is a local topology based method, the self–training issue could be avoided when a function estimation based method is employed to estimate the label of the unlabeled data. The parameter, k, was one that makes PLR sensitive. In order to overcome that issue, 2–PLR were employed. PLR_{local} captured the local topology of the unlabeled data while PLR_{global} captured the global label distribution. The final output was constructed by conjugating both PLRs' outputs. Then, the training data were generated from the unlabeled data and their estimated label distributions. Those unlabeled data with high uncertainty have high probability to be generated in order to generate more information. On the other hand, those unlabeled data with low uncertainty

have low probability to be generated in order to avoid redundancy. Every unlabeled data have same chances of trials, and in each trial, the σ_i^2 determines whether the unlabeled data point, $U_{\mathbf{x}_i}$, was generated or not. Since the proposed method generated more training data than the original training data, the data selection method, MDS, was employed to reduce the training complexity.

The experiments conducted on 18 datasets, and COREG$_{kNN}$, COREG$_{SVR}$ and Co–SVR were employed to the performance evaluation of the proposed method. The ratio of the labeled data was varied to 20%, 10%, 5% and 1% of original training dataset. The proposed method outperforms all the SVR re–training methods, COREG$_{SVR}$ and Co–SVR. The accuracy of the proposed method was about 1.5% higher than COREG$_{kNN}$, but very comparable. However, the training time of the proposed method was only 20∼25% of training time of COREG$_{kNN}$. The accuracy was stable to the parameter, t.

There are some limitations and future works of current work. Fist, some unlabeled data need to be removed when training the final SVR model. For SS–SVR, the uncertainty was used for determining the data generation rate. However, estimated labels of some unlabeled data may be too uncertain to be used for training. In that case, those data should be rejected to train the final regression model. Co–training based methods tend to reject the unlabeled data which are not upgrade the model accuracy. An unlabeled data point rejection step should be considered to avoid training the unlabeled data with arbitrary target values. Second, this approach can be applied to other regression methods. The future research direction can be the generalization of this work to other regression model based SSL.

Chapter 5

Application 1: Data Selection for Response Modeling

5.1 Response Modeling

A response model identifies customers who are likely to respond and the amount of profit expected from each customer using customer databases consisting of demographic data and purchase history for the purpose of direct marketing. With this model, marketers are able to decide who to contact within a limited marketing budget. A well–targeted response model can increase profit, while a mis–targeted response model not only decreases profit but also worsens the relationship between the company and customers (Blatberg et al., 2008; Gönül et al., 2000; Shin and Cho, 2006).

Various learning methods have been applied to response modeling. Logistic regression based response models (Hosmer and Lemeshow, 2000; Sen and Srivastava, 1990) and decision tree based response model (Haughton and Oulabi, 1997) have been proposed with its simplicity, explainability (Shin, 2005). For more complex methods, NN–based response modeling (Ha et al., 2005), SVM–based response modeling (Shin and Cho, 2006), 1–SVM based response modeling (Lee and Cho, 2007) and SSL–based response modeling (Lee et al., 2010) have been proposed. Those methods were employed a classification method in order to identify respondents. On the other hand, the importance of feature selection was reported in Malthouse (1999) and Malthouse (2002). The accuracy of a response model

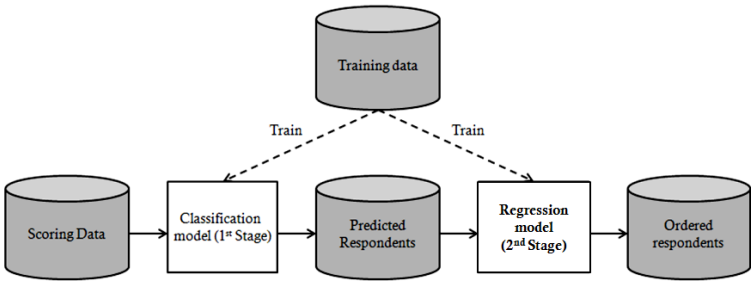


Figure 5.1: Concept of the two–stage response model.

can be improved when a proper feature set is selected and used for training. Another issue of the response model is the class–imbalanced problem. Since the response rate of a marketing campaign is very low, most response modeling dataset consist of a few respondents and many non–respondents. Class–imbalanced problem occurs when the number of data of a specific class is greater than the number of data of the other class. Since the majority class is overestimated, the normal binary classifier cannot construct the ideal class boundary. In order to overcome the class–imbalanced problem, under sampling, over sampling, ensemble and one–class learning have been widely used (Chawla et al., 2004).

5.2 Two–Stage Response Modeling

Usually, a response model employs a classification model to predict the likelihood to respond of each customer. Then, those likelihoods are directly used to sort the predicted respondents. However, as pointed out in KDD98 [1], there may be an inverse correlation between the likelihood to respond and the dollar amount to spend to some marketing datasets (Kim et al., 2008; Wang et al., 2005). In this case, profit may not be maximized because some low–spending customers are top–ranked, while some high–spending customers may be low–ranked. This is because the more dollar amount is involved, the more cautious a customer becomes in making a purchase decision. Therefore, an additional effort to maximize the profit should be added to the conventional response modeling.

[1] KDD98: `http://www.kdnuggets.com/meetings/kdd98/kdd-cup-98.html/`

Two-stage response modeling (See Figure 5.1), identifying respondents at the first stage and then ranking them according to expected profit at the second stage, was proposed to overcome this problem (Kim et al., 2008). In the first stage, conventional classification response models can be directly applied to predict desirable respondents based on their likelihood to respond. However, for the second stage, a new model is needed to estimate the purchase amount of respondents. SVR is one possible solution for use in the second stage of the two-stage response modeling with its ability of solving nonlinear problems.

The training complexity problem is still an issue for two–stage response modeling. Response modeling datasets usually consist of very large training data, sometimes including billions of transactions from millions of customers. In addition, data analysis for a marketing campaign includes the construction of various models with different samples of a dataset to verify multiple marketing actions. Moreover, SVR contains an additional hyper–parameter which requires that the SVM classifier, ε, be set empirically. Hence, response modeling with SVR consists of a repeated modeling process with a very large dataset and including parameter searching processes. The training complexity of SVR must be reduced for use in practical two-stage response modeling.

5.3 Experimental Results

5.3.1 Experiment Setting

In Section 5.3, the experimental profit results of a real–world marketing dataset, The Direct Marketing Educational Foundation 4 (DMEF4) dataset[2], is presented. The DMEF4 contains 101,532 customers and 91 input variables with a 9.4% response rate for a donation mailing task. Only 15 relevant input variables were used for training, as shown in Table 5.1, following previous research (Ha et al., 2005; Malthouse, 2002; Yu and Cho, 2006). For performance evaluation, we constructed ten random datasets following the procedure of Lee and Cho (2007). The original dataset was randomly partitioned in half: the training set and the test set. Each dataset had 50,766 training data including 4,876 respondents and 50,766 test data including 4,875 respondents. All performances were measured by averaging the experimental results of these ten datasets.

[2]DMEF Dataset: `http://///www.directworks.org/academics/`

Table 5.1: Input variables of DMEF4 dataset.

Name	Formulation	Description
ORIGINAL VARIABLES		
Purseas		Number of seasons with a purchase
Falord		LTD fall orders
Ordtyr		Number of orders this year
Puryear		Number of years with a purchase
Sprord		LTD spring orders Derived Variables
DERIVED VARIABLES		
Recency		Order days since 10/1992
Tran53	$I(180 \leq \text{recency} \leq 270)$	
Tran54	$I(270 \leq \text{recency} \leq 366)$	
Tran55	$I(366 \leq \text{recency} \leq 730)$	
Tran38	1/recency	
Comb2	$\sum_{m=1}^{14} \text{ProdGrp}_m$	Number of product groups purchased from this year
Tran46	$\sqrt{\text{comb2}}$	
Tran42	$\log(1 + \text{ordtyr} \times \text{falord})$	Interaction between the number of orders
Tran44	$\sqrt{\text{ordhist} \times \text{sprord}}$	Interaction between LTD orders and LTD spring orders
Tran25	1/(1+lorditm)	Inverse of latest-season items

For the first stage of response modeling, we implemented two different classification models, 1–class SVM (1–SVM) and 2–class SVM (2–SVM), both of which have been successfully applied to response modeling with great generalization performances. For 1–SVM, only respondent data were trained. For 2–SVM, both respondent and non–respondent data were trained, while the undersampling method was used to overcome the class–imbalanced problem. The performance of a response model was measured by averaging over ten undersampling experiments because of the randomness. Then, for the second stage of response modeling, SVR and SVR with MDS was employed. For the original SVR, all respondent data with positive actual purchase amounts were employed. For SVR with MDS, MDS selected the important data from the original dataset and the selected set were trained by SVR.

The hyper-parameters of 1–SVM, 2–SVM and SVR were set using ten fold cross–validation, while an RBF kernel with σ fixed at 1.0 was employed for all models. The control parameter of MDS, α, was varied as 0, 0.5 and 1, while l and m were set to 10 and 20% of original dataset. All experimental results with MDS were averaged over ten repetitions, and all data were normalized.

Table 5.2: Experimental performance of classification models.

Classification Method	BCR	ROC Distance
1–SVM	54.89%	66.06%
2–SVM	71.40%	30.79%

5.3.2 Experimental Results

Before analyzing the experimental results of response modeling, the performances of the classification models should be discussed. The measurements for a class–imbalanced problem, the Balanced Classification Rate (BCR) and ROC distance. The equation of BCR and ROC distance is depicted in Eq. 5.2 and Eq. 5.1,

$$BCR = \sqrt{\frac{TP}{N_R} \times \frac{TN}{N_{NR}}}, \tag{5.1}$$

$$ROCDistance = \sqrt{\frac{FN}{N_R}^2 \times \frac{FP}{N_{NR}}^2}, \tag{5.2}$$

where TP, TN, FN, FP, N_R, N_{NR} indicate True Positive, True Negative, False Negative, False Positive, the number of respondents and the number of non–respondents, respectively. Hence, a larger BCR and smaller ROC Distance guarantees a better performance. As shown in Table 5.2, 2–SVM outperformed 1–SVM because 2–SVM includes both respondents and non–respondents into account, and the undersampling method upgraded the performance. However, both experiments showed comparable results to those in other studies on response modeling with real–world marketing datasets.

Figure 5.2 shows the experimental results of response models based on 1–SVM in terms of actual profit per mail. The X–axis indicates the ordered decile, sorted in terms of model score, while the Y–axis indicates the corresponding actual profit. The mailing costs varied from \$1, \$3, \$5 and \$10, "1–SVM only" indicates the experimental result from a response model using only 1–SVM, which is a conventional response model, while "SVR only"' indicates the experimental result from a response model with only SVR. We also included a previous version of this function, "Hybrid Score" (Kim and Cho, 2010) for comparison. For "1–SVM + SVR," we selected

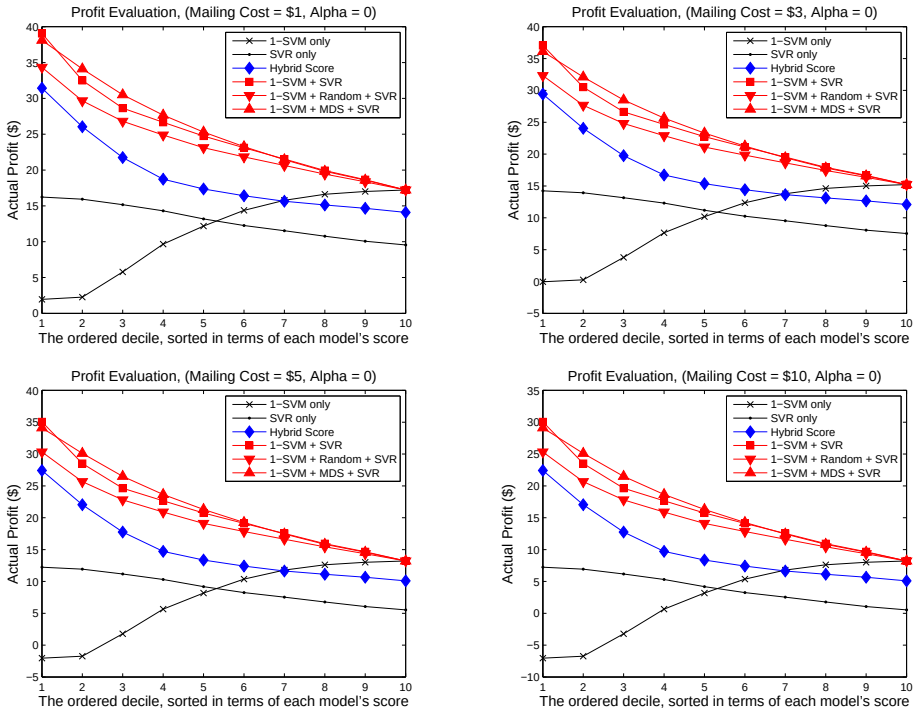


Figure 5.2: Experimental results of response models based on 1–SVM.

respondents with 1–SVM and ranked them by predicted purchase amount from SVR. We also employed MDS to reduce the number of training data of SVR, "1–SVM + MDS + SVR," and employed random sampling to compare the data selection performance, "1–SVM + Random + SVR." As shown in Figure 5.2, there were strong inverse correlations between the 1–SVM score and the actual profit. On the other hand, the two–stage response models showed higher profits than did the conventional response model, while "1–SVM + MDS + SVR" showed comparable results. The profit earned from "1–SVM + MDS + SVR" was almost the same as the profit from the model without MDS, while "1–SVM + Random + SVR" had an inferior profit.

Figure 5.3 shows the experimental results of response models based on 2–SVM in terms of actual profit. Similar to the results based on 1–SVM, there was no linear correlation between 2–SVM output and actual profit. The results showed that the two–stage response model was es-

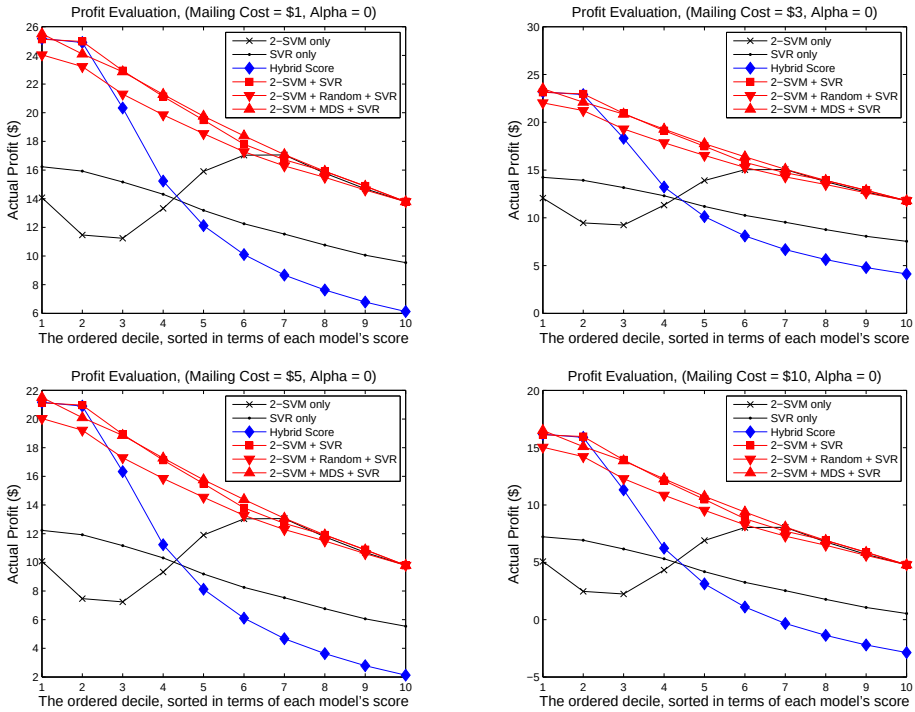


Figure 5.3: Experimental results of response models based on 2–SVM.

pecially suitable for identifying top–ranked respondents. Similar to the results of 1–SVM, the performances of "2–SVM + MDS + SVR"'' are very comparable to those of the original SVR.

Figure 5.4 shows the reduced training time complexity due to MDS compared to the complexity of the original SVR for the ten training datasets used in Figure 5.2 and Figure 5.3. For the ten training datasets, the training time complexities decreased an average of 57% of that of the original SVR. With those results, it seems that SVR employing MDS results in profit almost the same as that of the original SVR while using only 57% of the training time. We concluded that MDS can be used for the efficient training for SVR with minimum loss of accuracy.

Figure 5.5 shows the effect of the MDS control parameter α. The ratios of the RMSE of SVR with MDS to the RMSE of SVR without MDS are plotted for ten training sets. An α of 1 results in the best and most stable training accuracy. A smaller α results in suboptimal training accuracy but

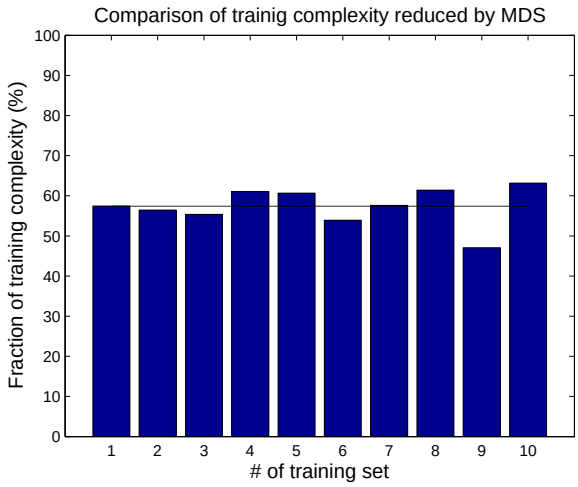


Figure 5.4: Training complexity reduced by MDS compared to that of the original SVR.

more efficient training. This demonstrates that α controls the trade–off between model accuracy and training efficiency. We suggest that a smaller α is useful in parameter search or early stages of response modeling, while a higher α is useful for final customer–targeting stages.

5.4 Summary

Support Vector Regression (SVR) has been employed for response modeling. One drawback to this method is its relatively high training complexity. Since the training complexity of SVR is highly correlated to the size of training dataset, and response modeling usually includes large marketing datasets, the data selection approaches are preferred. In this paper, we proposed the Margin based Data Selection method (MDS) to reduce the training complexity of SVR for two–stage response modeling. MDS selects only those data that are likely to become support vectors and automatically determines the number of data selected. This property is essential for obtaining good results and reducing the parameter search time which is time–consuming and requiring experiences for both the training algorithm and the dataset characteristic. The parameter α controls the number of data selected by the model, allowing for adaptation to various

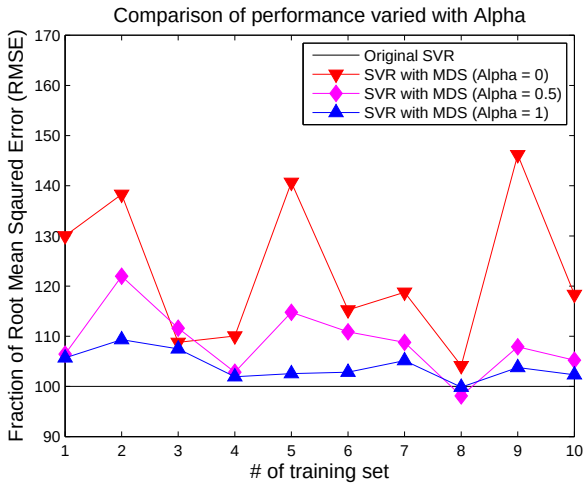


Figure 5.5: Comparison of original SVR and SVR employing MDS to various Alpha.

noise levels or different marketing stages. Through the experiments including 20 datasets, we showed that MDS provided better generalization performances than did the other benchmark methods regardless of their characteristics of artificial, time series, non–time series, linear, non–linear, low noise or high noise. For the experiments using a real–world marketing dataset, we implemented both the first and the second stage of response modeling involving 1–SVM, 2–SVM, SVR and MDS. The experimental results showed that MDS successfully reduced the training complexity of SVR for response modeling with minimum loss of accuracy.

Chapter 6

Application 2: Semi-Supervised Support Vector Regression for Virtual Metrology

6.1 Virtual Metrology

In semiconductor manufacturing, a wafer needs to be processed by hundreds of different manufacturing processes, such as photolithography (photo) and etching. In a manufacturing process, a wafer containing thousands of semiconductors is processed according to recipes. The quality measurement of a semiconductor product is yield, which indicates how many semiconductors activate in a wafer. However, yield determined after all of the processes cannot detect faulty wafers occurred in the process. If such faulty wafers remain during subsequent processes, they increase the manufacturing cost and production lead time. Hence, an additional quality management measurement for the early detection of faulty wafers is needed.

In order to detect faulty wafers earlier, a metrology process is employed after each manufacturing process (Kang et al., 2009). In the metrology process, metrology equipment inspects the wafer quality indicators, such as the critical dimension of the etching process or the axes distortion of the photo process. If the metrology value is within the pre–defined metrology thresholds, then the wafer is considered as normal, otherwise the wafer is deemed to be faulty (Kourti and MacGregor, 1995; Qin, 2003; Qin et al.,

2006; Su et al., 2007). However, the actual metrology process requires extra cost, increased human resources and a longer cycle time (Chang et al., 2006; Cheng and Cheng, 2005). Hence, only one wafer per a process lot of 25 wafers is sampled for inspection and the remaining 24 wafers are not inspected at all.

Many efforts have been made to overcome this limitation. One example is statistical process control (SPC) based on the fault detection and classification (FDC) data. In semiconductor manufacturing, each piece of processing equipment contains hundreds of sensors to monitor process conditions, such as temperature, pressure and the process time. FDC data refer to those observations directly from the sensors in the processing equipment. Univariate SPC inspects each FDC variable independently and decides that a wafer is faulty when an FDC variable is located outside the pre–defined lower control limit (LCL) and upper control limit (UCL). Multivariate SPC uses a dimensionality reduction method, such as principal component analysis (PCA), to condense FDC variables into one explainable variable, and then classifies wafers based on that explainable variable. However, FDC–based SPC has critical limitations. First, since FDC data are originally collected to monitor the process conditions, the critical variables that affect the quality of the wafers are not known. In addition, the range of values for each variable that makes a wafer faulty is also not known. Moreover, since FDC data have high–dimensional and high variance variables, the conventional SPC cannot detect nonlinear correlations and intersections among those variables.

To overcome the limitation of FDC–based SPC, virtual metrology (VM) has been proposed to model the relationships between FDC data and metrology values (Chen et al., 2005). VM refers to "the estimation of metrology values based on process data such as fault detection and classification (FDC), context and previous metrology." (Besnard and Toprac, 2006) The bottom of Figure 6.1 illustrates the concept of VM, while the top depicts the actual metrology. Since VM can employ nonlinear models such as NN and SVM, VM has an advantage in that it considers the nonlinear relationships between FDC variables and the metrology variable. In addition, since VM predicts the metrology values of all wafers, more accurate quality management is possible without increasing production lead time. Based on those advantages, the VM concept has been successfully applied to run–to–run (R2R) control and faulty wafer detection systems (Kang et al., 2011; Kim et al., 2012). For application to fab–wide semicon-

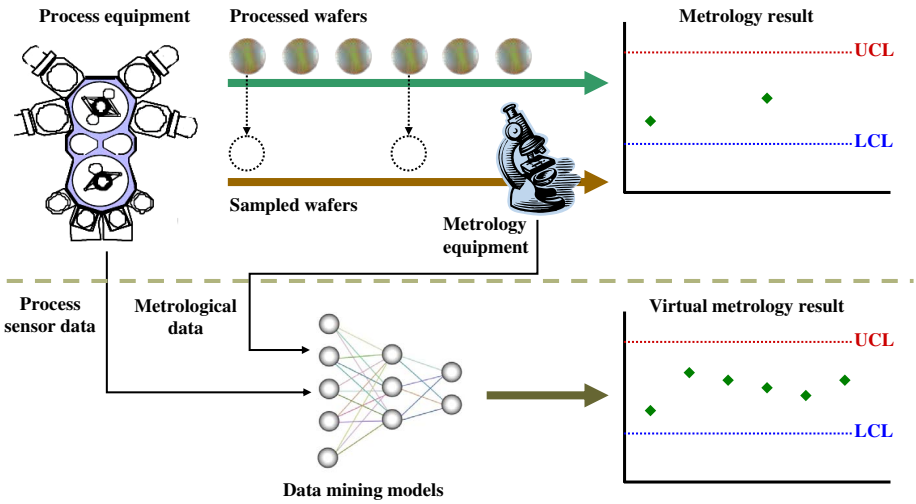


Figure 6.1: Concept of the actual metrology and the virtual metrology (Kang et al., 2009).

ductor manufacturing processes, a more accurate VM model is required. A less accurate VM model will decrease the manufacturing performance.

6.2 Semi–Supervised Learning for Virtual Metrology

Supervised regression methods, such as Multivariate Linear Regression (MLR) (Johnson and Wichern, 1998), k–NN regression (Bishop, 2006; Duda et al., 2001) and Neural Networks (NN) (Haykin, 1999) have been employed for training a VM model. Recently, SVR–based VM model is widely used with its great performances (Kang et al., 2011, 2009; Kim et al., 2012). The conventional supervised VM model used only labeled data to train. Since all processed wafers have FDC data, the input data are very plenty. However, only one out of 25 wafers is sampled for inspection by the metrology step. Hence, only $\frac{1}{4}$ of the input data have their corresponding target data (Figure 6.2). Other $\frac{3}{4}$ data exist without target data. This nature of VM makes it a good application for SSL. In order to upgrade the performance of VM, the SS–SVR method proposed in Chapter 4 is applied with a real–world dataset.

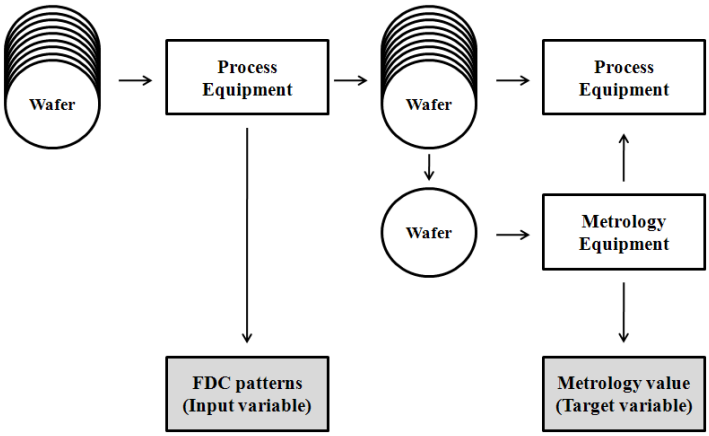


Figure 6.2: Input and target variables of VM.

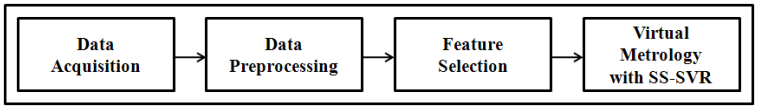


Figure 6.3: Process steps the virtual metrology.

6.3 Virtual Metrology Process

6.3.1 Overview

Figure 6.3 indicates the VM process steps. Data acquisition and preprocessing steps were conducted first. Then, features were selected for VM modeling from among hundreds of original input variables, i.e., FDC variables. Finally, VM modeling including SSL step was conducted with the constructed VM training dataset so that the effect of SS–SVR could be evaluated.

6.3.2 Data Acquisition

The dataset used in this research was collected during a photolithography (photo) process at a Korean semiconductor manufacturing company over four months. During the photo process, the quality of the wafer can be decreased by axis rotation and distortion, etc. FDC data were acquired from two pieces of processing equipment (EQ1 and EQ2) and metrology

Table 6.1: VM dataset.

Process equipment	Period	# of the labeled data	# of the unlabeled data	# of FDC variables (input variables)
EQ1	P1	230	3670	117
	P2	172	2872	117
	P3	137	2313	112
	P4	167	3870	112
	P5	452	6546	112
	P6	818	9929	112
	P7	138	1654	112
	P8	195	2325	112
EQ2	P1	226	3377	117
	P2	180	2639	117
	P3	136	2293	112
	P4	170	3398	112
	P5	450	6008	112
	P6	816	9100	112
	P7	138	1523	112
	P8	195	2132	112

data from metrology equipment after the photo process. 133 FDC variables and one corresponding metrology variable were collected.

6.3.3 Data Preprocessing

Seven preventive maintenance (PM) processes took place during the period of data collection. PM causes significant changes in the recipe and the sensors of the processing equipment. Hence, the original dataset was partitioned into eight datasets (P1–P8) by PM periods to construct independent VM models for each instance of PM. After that, some FDC variables, which are not needed for modeling, such as wafer ID and process ID, were deleted. All datasets used are summarized in Table 6.1.

6.3.4 Feature Selection

As shown in Table. 6.1, FDC variables 112–117 differed by period. However, those FDC variables were originally collected for process monitoring by sensors in the processing equipment, not for VM modeling. Hence, we had to select features that are useful for training VM models (Kang et al., 2011, 2009; Kim et al., 2012). If all original features are used, then the model complexity increases while overfitting occurs. In this research, we employed the genetic algorithm (GA) for feature selection (Mitchell, 1996; Yang and Honavar, 1998). Since the focus of this chapter is SS–SVR based

Table 6.2: The number of selected features.

Equipment	Period	# of selected features (GA–SVR)
EQ1	P1	14
	P2	11
	P3	5
	P4	9
	P5	12
	P6	8
	P7	2
	P8	5
EQ2	P1	8
	P2	8
	P3	2
	P4	10
	P5	7
	P6	7
	P7	8
	P8	6

VM, SVR was used as the base model for GA (GA–SVR) with only labeled data. The number of selected features are summarized in Table. 6.2.

6.3.5 Virtual Metrology Modeling

After all preprocessing steps including feature selection, the VM model was trained. For the conventional VM model, supervised regression methods, such as MLR and SVR, have been usually employed. However, contrast to the conventional VM model, SSL based VM models were employed in order to upgrade the accuracy of VM. In addition, in order to see whether the performance was upgraded or not by SSL, a VM model trained only labeled data was employed.

6.4 Experimental Results

6.4.1 Experiment Setting

For benchmark methods, COREG$_{kNN}$, COREG$_{SVR}$ and Co-SVR, which were once used in the experiment in Chapter 4, were employed. Those co–training methods have parameters to set. The size of the working set for the unlabeled data was set to 200 for COREG and was set to 40 for Co-SVR. The maximum number of iterations was set to 100 for all methods. The number of nearest neighbors for COREG was set to 5. Each initial

training set for COREG had all labeled data while the unlabeled data was partitioned into each initial training set for Co-SVR, as described in Wang et al. (2010); Zhou and Li (2007). For the proposed method, the number of nearest neighbors for 2–PLR were set to 5 and 20 for k_{local} and k_{global}, respectively. The number of trails for each unlabeled data in data generation, t, is set to 3, 5. The hyper–parameters of SVR were determined by cross–validation. RBF kernel was used as a kernel function and the kernel parameter σ was fixed to 1.0 for all datasets. All datasets were normalized. The features were selected by GA. The number of populations, the number of maximum iterations, the crossover rate and the mutation rate were set to 100, 300, 0.5 and 0.03, respectively.

Five–fold cross–validation was used for the performance evaluation. The labeled data partitioned into five folds, randomly, and a model was trained with the unlabeled data with the labeled data from four folds. The labeled data of the other fold was set to be the test data. The RMSE was selected for performance measurements. RMSE measures the average of the difference between the actual target value and the estimated target values by VM, which is commonly used in real semiconductor manufacturing.

6.4.2 Experimental Results

Figure 6.4 and Figure 6.5 show the experimental results on EQ1 and EQ2, respectively. The pairs of RMSE and training time in seconds are plotted corresponding to each method. The closer a result is plotted to the origin, the better the method performs. The benchmark methods, COREG$_{kNN}$, COREG$_{SVR}$ and Co-SVR were plotted as upper triangles, lower triangles and diamonds, respectively. The performances of the proposed method varied by the parameter t were plotted as blue squares and connected with a line. The experimental results of the proposed method were not affected by the parameter t, except P1 of EQ1, P5 and P8 of EQ2. However, even in those cases, the accuracy were not sensitive to t. The performances of the proposed method were comparable to COREG$_{kNN}$ or COREG$_{SVR}$ while the proposed method outperformed Co-SVR.

For more details, Table 6.3 and Table 6.4 depict the RMSE results and the training time of the experimental results on EQ1, respectively. The training dataset differed by training period, P1-P8, each of which was considered as a independent dataset. COREG$_{kNN}$, COREG$_{SVR}$ and Co-SVR are the benchmark methods. “Proposed, 3” and “Proposed 5”

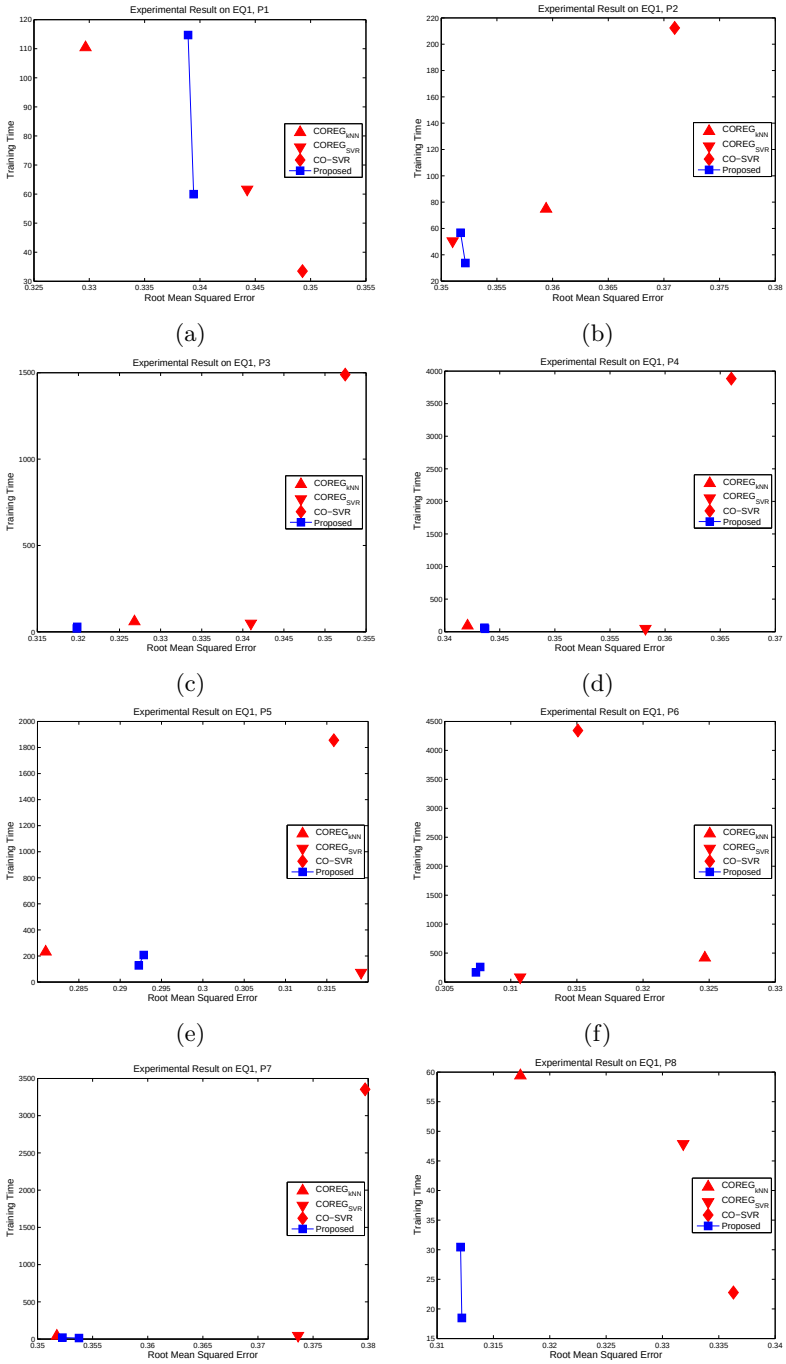


Figure 6.4: Experimental results on EQ1.

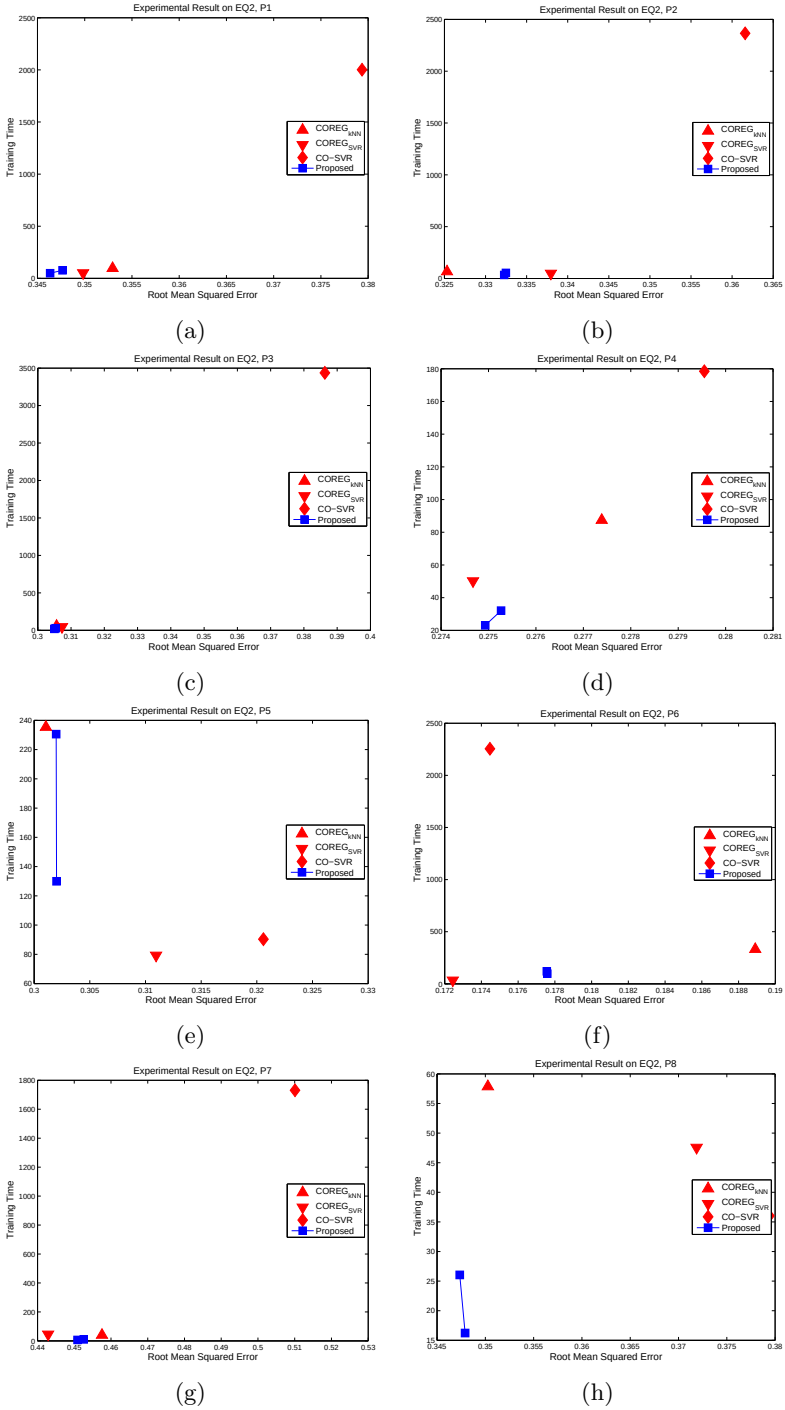


Figure 6.5: Experimental results on EQ2.

Table 6.3: RMSE of the experimental results on EQ1.

	Labeled	COREG$_{kNN}$	COREG$_{SVR}$	Co-SVR	Proposed, 3	Proposed, 5
P1	0.3461	**0.3297**	0.3443	0.3493	0.3394	0.3389
P2	0.3543	0.3594	**0.3510**	0.3710	0.3522	0.3517
P3	0.3547	0.3268	0.3410	0.3525	**0.3198**	0.3199
P4	0.4056	**0.3421**	0.3582	0.3660	0.3437	0.3436
P5	0.3191	**0.2810**	0.3191	0.3159	0.2923	0.2929
P6	0.3167	0.3247	0.3107	0.3151	**0.3074**	0.3077
P7	0.4021	**0.3518**	0.3736	0.3797	0.3538	0.3523
P8	0.3318	0.3174	0.3318	0.3363	0.3122	**0.3121**
Avg.	0.3538	0.3291	0.3412	0.3482	0.3276	**0.3274**

Table 6.4: Training time of the experimental results on EQ1.

	COREG$_{kNN}$	COREG$_{SVR}$	Co-SVR	Proposed, 3	Proposed, 5
P1	110.43	61.61	33.51	59.97	114.71
P2	74.86	50.41	212.34	33.76	56.73
P3	59.43	48.87	1489.20	19.50	28.93
P4	91.18	47.24	3884.27	43.76	60.60
P5	231.89	72.48	1855.78	127.53	207.39
P6	418.72	83.05	4342.64	166.13	260.26
P7	41.98	46.36	3353.78	13.89	19.60
P8	59.41	47.87	22.77	18.48	30.44
Avg.	135.99	57.24	1899.29	60.38	97.33

indicate the proposed method with the different parameter, $t = \{3, 5\}$, respectively. As shown in Table 6.3, COREG$_{kNN}$ outperformed for four datasets while the proposed method outperformed for three datasets in terms of RMSE. However, "Proposed, 5" was the best on the average of RMSE and "Proposed, 3" was the second. Even COREG$_{kNN}$ was best for some datasets, it may not be stable for all datasets. Table 6.4 shows the training time of each method. The proposed method was faster than co–training based benchmark methods, except COREG$_{SVR}$. Both "Proposed, 3" and "Proposed, 5" showed better efficiency than COREG$_{kNN}$ and Co-SVR. The reason that COREG$_{SVR}$ used less time than COREG$_{kNN}$ is that the interpolation problem occurred by the single target value of the unlabeled data, which makes SVR use a few numbers of support vectors. Hence, the accuracy of COREG$_{SVR}$ was worse than COREG$_{kNN}$ and the proposed method.

Table 6.5 and Table 6.6 depict the RMSE results and the training time of the experimental results on EQ2, respectively. COREG$_{SVR}$ and the proposed method outperformed three datasets, each. However, "Proposed, 3" was the best on the average of RMSE and "Proposed, 5" was the second.

Table 6.5: RMSE of the experimental results on EQ2.

	Labeled	COREG$_{kNN}$	COREG$_{SVR}$	Co-SVR	Proposed, 3	Proposed, 5
P1	0.3602	0.3529	0.3498	0.3793	**0.3463**	0.3476
P2	0.3502	**0.3253**	0.3380	0.3616	0.3323	0.3325
P3	0.3497	0.3057	0.3073	0.3863	**0.3050**	0.3054
P4	0.2709	0.2774	**0.2747**	0.3796	0.2749	0.2753
P5	0.3136	**0.3011**	0.3110	0.3206	0.3020	0.3020
P6	0.1727	0.1889	**0.1724**	0.1745	0.1776	0.1776
P7	0.4772	0.4575	**0.4429**	0.5101	0.4509	0.4526
P8	0.3719	0.3503	0.3719	0.3794	0.3479	**0.3473**
Avg.	0.3331	0.3199	0.3210	0.3489	**0.3171**	0.3175

Table 6.6: Training time of the experimental results on EQ2.

	COREG$_{kNN}$	COREG$_{SVR}$	Co-SVR	Proposed, 3	Proposed, 5
P1	96.99	51.60	2001.86	49.62	77.01
P2	66.65	48.84	2365.47	33.92	53.94
P3	56.55	46.22	3436.35	19.77	28.05
P4	87.44	5018	178.48	23.10	32.10
P5	235.37	79.27	90.32	129.94	230.56
P6	332.21	33.93	2255.14	97.46	120.34
P7	40.27	45.16	1730.96	6.70	9.80
P8	57.87	47.57	36.03	16.23	26.05
Avg.	121.67	50.34	1511.83	47.09	72.23

COREG$_{SVR}$ showed worse than COREG$_{kNN}$ in terms of the average of RMSE. The proposed method may be much stable than COREG method. Table 6.6 showed the training time of the experimental results on EQ2. "Proposed, 3" showed the best training time, while Co-SVR showed the worst. Based on Table 6.3-Table 6.6, the proposed method showed the best RMSE for various datasets on average. The training time of the proposed method, differed by the parameter t, was short than the training time of COREG$_{kNN}$ and Co-SVR. Consequently, the proposed method showed the most efficient experimental results for various datasets in terms of RMSE and the training time.

Figure 6.6 illustrated RMSE ratio compared to the conventional VM model which trains the labeled data only. The improvement of RMSE by using SSL methods can be measured in Figure 6.6. As the results, the proposed method can improve the conventional VM model for 5-8% in terms of RMSE which is the best results compared to benchmark methods. For the VM, where a small improvement can upgrade the quality of wafers, those experimental results showed that the proposed method can improve the conventional VM model.

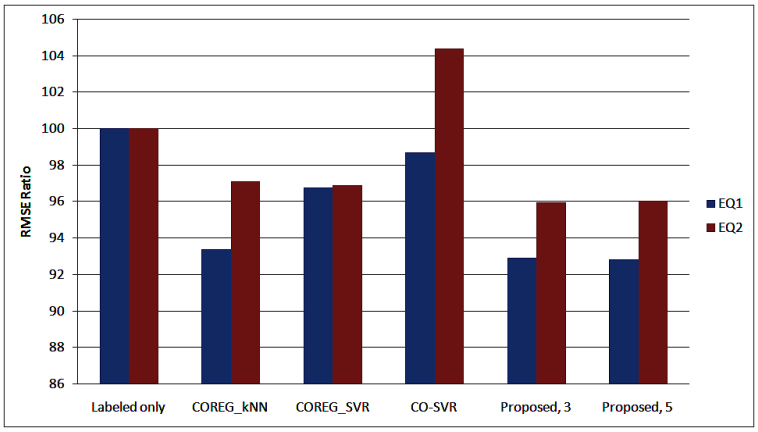


Figure 6.6: RMSE ratio compared to train the labeled data only.

6.5 Summary

In semiconductor manufacturing, quality management is a key issue. Since the main quality measurement, yield, is gathered at the end of the manufacturing process, the metrology process is applied after each individual process for the early detection of faulty wafers to save the manufacturing cost. However, the metrology process only measures one sample wafer out of a lot consisting of 25 wafers. To overcome that problem, VM is proposed. VM estimates the metrology values for all wafers by modeling the relationship between the FDC variables and the metrology variables. The performance of VM determines the success of the quality management for semiconductor manufacturing.

The conventional VM model employs a supervised regression method in order to identify the relationship between the FDC variables and the metrology value. Since metrology values of the labeled data for VM is obtained by actual metrology steps, there exist a lot of the unlabeled data. The SSL regression, which trains both labeled and unlabeled data, can improve the performance of the conventional VM model.

In this Chapter, the proposed SS–SVR method in Chapter 4 was applied to the semi–supervised VM problem. The proposed method employed 2–PLR for labeling the unlabeled data. Through the data generation, information for the uncertain region could be generated and data were located for training the maximum margin model. In order to reduce the training

complexity, EMPS from Chapter 3 was employed. The experiments conducted on a real–world semiconductor manufacturing dataset collected by two pieces of equipment (EQ) over four months at a Korean semiconductor manufacturing company. The metrology values for the labeled data were collected from a metrology machine. The original dataset was divided into eight training periods based on PM. A GA–based wrapper approach was employed for feature selection. Then, the SSL regression methods including the proposed method were employed to train a VM model. COREG$_{kNN}$, COREG$_{SVR}$ and Co-SVR were employed for benchmark methods. The experimental results showed that the proposed method were the most efficient method among the benchmark methods. RMSE of the proposed method was comparable to COREG methods. However, the training time of the proposed method was better than the training time of COREG. In addition, the average RMSE of the proposed method was the best, which means the proposed method was stable to various datasets. The proposed method can improve the accuracy of the conventional VM by 5-8%. Semiconductor processing units are getting smaller and the details in manufacturing recipes are getting more important, and the proposed method is able to make the semiconductor manufacturing process more accurate and more efficient.

This approach has limitations that should be addressed in future work. First of all, the original training dataset were divided into eight datasets based on PM in order to train individual models for each training period. Further studies should aim to automatically adapt to the major changes in data trends. Second, this approach can be applied to other semiconductor manufacturing processes besides the photo process.

Chapter 7

Conclusion

7.1 Summary and Contributions

In this dissertation, two major open problems for SVR were considered: (1) reduction of the training complexity and (2) utilization of the unlabeled data for SS–SVR.

- **Training complexity**
  The training complexity of SVR is too high to train a large number of data. The training complexity of SVR is highly related to the number of training data n: $O(n^3)$, training time complexity and $O(n^2)$, the training memory complexity. Since the number of training data increases in the big data generation, the training time of SVR is expensive, and occasionally, SVR does not work in a limited memory space for large datasets.

  In order to reduce the training complexity, a data selection method, MDS, was proposed. Since the training complexity of SVR is highly related to the number of training data, reducing he number of training data is an effective approach to overcome the training complexity problem. Data selection approach is designed to select important or informative data among all training data. The goal of data selection is to reduce the training complexity as low as possible while retaining the accuracy as high as possible. For SVR, the most important data are support vectors which are the components of linear combination for a constructed SVR regression function. Since the support vectors are

selected from training data during the SVR training, before training, it is impossible to identify support vectors. MDS used a geometrical characteristic of support vectors. By ε–loss foundation and the maximum margin learning, all support vectors of SVR are located on or outside the ε–tube. That means the margin of support vector are equal or greater than the predefined parameter, ε. With multiple sample learning, MDS estimated the expected margin for all training data, efficiently. Those training data, whose expected margin is equal or greater than ε, were selected to train the final SVR. Through the experiments conducted on 20 datasets, the performance of MDS was better than the benchmark methods. The training time of SVR including running time of MDS was with 38% $\sim$ 67% of training time of original datasets. At the same time, the accuracy loss was 0% $\sim$ 1% of original SVR model.

- **Semi–supervised SVR (SS–SVR)**
  Recently, the size of dataset is getting larger and data are collected from various applications. Data can be divided into two groups: the labeled data and the unlabeled data. Since collection of the labeled data is expensive and time consuming, the fraction of the unlabeled data is getting increased. The conventional supervised learning method uses only labeled data. Many SSL methods have been proposed in order to improve the conventional supervised learning methods by using the unlabeled data along with the labeled data. However, since the target variable is a continuous variable, SSL regression is more tricky than SSL classification.

  Co–training based methods were the state–of–the–art for the SSL regression. Co–training has some drawbacks. First, the training complexity of co–training is relatively high. Co–training is an iterative method and construct u models for each iteration where u is the number of the unlabeled data. Moreover, the number of training data is getting larger by adding the unlabeled data as the iteration goes. Second, the uncertainty of estimating the labels of the unlabeled data is not considered. For some unlabeled data, the estimation of the corresponding label may not be correct. Finally, co–training method uses single label for the unlabeled data, which may occur an interpolation problem. Since SVR is a maximum margin learning method, the unlabeled data with single label value may not give new information of the underlying function.

  In this dissertation, a data generation and selection method for SS–SVR

training was proposed. In order to estimate the label distribution of the unlabeled data, a probabilistic local reconstruction method, PLR, was employed. Label distribution of the unlabeled data have two advantages than label value itself. First, the uncertainty can be estimated. The standard deviation of the estimated label distribution represents the uncertainty of estimation for the unlabeled data. Second, the information of a region that label value can located can be obtained rather than a single label value. Two PLR were employed in order to stable to noisy data, and the final label distribution was obtained by the conjugation of 2–PLR. Then, the data generation step was employed. With the estimated label distribution, training data were generated from the unlabeled data. The number of data generated was differed by the uncertainty. For those unlabeled data with high uncertainty, the data generation rate is relatively high in order to obtain information for the uncertain region. On the other hand, for those unlabeled data with low uncertainty, the data generation rate is relatively low in order to avoid redundancy. After that, MDS was employed to reduce the training complexity increased by the generated data. Through the experiments conducted on 18 datasets, the proposed method showed good results. The proposed method could improve about 10% of the accuracy than the conventional supervised SVR, which is comparable results to benchmark methods. At the same time, the training time of the proposed method including the construction of final SVR was less than 25% of benchmark methods. As consequence, the proposed method can improve the conventional supervised SVR using the unlabeled data with a minimum additional time. The experimental results were rarely affected by the parameter t.

For the applications, real–world datasets were employed. A response modeling dataset for response modeling was employed for a marketing application while a virtual metrology dataset for semiconductor manufacturing was employed for a manufacturing application.

- **Response modeling application**
  A response model identifies customers who are likely to respond and the amount of profit expected from each customer using customer databases consisting of demographic data and purchase history for the purpose of direct marketing. Usually, a response model employs a classification model to predict the likelihood to respond of each customer. However the classification response model may not maximize the profit of a response model.

In this dissertation, SVR based two–stage response modeling, identifying respondents at the first stage and then ranking them according to expected profit at the second stage, was proposed. Since response modeling datasets usually consist of very large training data, the training complexity problem is still an issue for SVR based two–stage response modeling. Hence, MDS was employed in order to reduce the training complexity of two–stage response modeling. One–class SVM and two–class SVM were employed for the first stage of two–stage response modeling. SVR with MDS was employed the second stage. The experimental results showed that SVR employed two–stage response model could increase the profit than the conventional response model. MDS reduced the training complexity of SVR to about 60% of original SVR with minimum profit loss.

- **Virtual metrology application**
  In semiconductor manufacturing, a wafer needs to be processed by hundreds of different manufacturing processes. A metrology process is employed after each manufacturing process for the quality management. However, since the actual metrology process requires extra cost, increased human resources and a longer cycle time, only one wafer per a process lot of 25 wafers is sampled for inspection and the remaining 24 wafers are not inspected at all. To overcome the limitation VM has been proposed to model the relationships between FDC data and metrology values. The conventional VM employs a supervised regression method with the labeled data having actual metrology values.

  In this dissertation, a SS–SVR method was applied to a real–world VM dataset by using the unlabeled data with the labeled data for training. Data were collected from two equipments of the photo process. Co–training based benchmark methods were employed to performance evaluation. The experimental results showed the proposed SS–SVR method could improve about 8% of average accuracy than the conventional VM model, which were comparable results to benchmark methods. The additional training time for the proposed method was relatively small than benchmark methods.

7.2 Limitations and Future Work

In this dissertation, methods for the training complexity of SVR and the training approach of SS–SVR have been proposed. Even the proposed methods showed a good performance by plenty experiments, there still exist future works. The limitations of the proposed methods and the direction of future works can be addressed as follows.

- **Data redundancy**
  MDS selects data which are likely to become support vectors by estimating margin of training data. This is a different view to others which selects a few representative data from a dense region. Hence, MDS tends to select too many data if the number of support vectors are too large or the parameter ε is too small. An additional effort that selecting representative data from a dense region to avoid redundancy can be employed to selected data by MDS.

- **Unlabeled data rejection**
  For SS–SVR, the uncertainty was used for determining the data generation rate. However, estimated labels of some unlabeled data may be too uncertain to be used for training. In that case, those data should be rejected to train the final regression model. Co–training based methods tend to reject the unlabeled data which are not upgrade the model accuracy. An unlabeled data point rejection step should be considered to avoid training the unlabeled data with arbitrary labels.

- **Parameter selection**
  MDS has two parameters: l, the number of multiple sample set and m, the number of data in a sample set. The proposed SS–SVR method has three parameters: k_{local}, the number of nearest neighbors for PLR_{local}, k_{global} the number of nearest neighbors for PLR_{global} and t, the number of trials for data generation. Though the experiments, some guidelines for those parameters can be obtained and they were effective. However, the fundamental bounds for parameters are not researched. Efforts to determination of the fundamental bounds are future research area.

- **Extension of SS–SVR for other regression models**
  The proposed SS–SVR method is basically designed for SS–SVR. The data generation step spreads training data in order to keep a margin area for maximum margin learning method, such as SVR. However, the idea

of SS–SVR proposed in this dissertation can be applied to other regression models, such as NN regression or k–NN regression. Of course, the data generation step including estimating the label distribution of the unlabeled data should be tuned for those Empirical Risk Minimization (ERM) based regression models or instance learning based regression models. Another research direction is the development of SSL regression framework for all regression models, rather than limited to SS–SVR.

- **Applications**
  A marketing application and a manufacturing application were experimented in this dissertation. However, more application can be experimented, such as telecommunications, SNS and mobile application usages. Especially, SNS has become an emerging area, and a lot of data including the unlabeled data has been generated. One research direction can be a development of a sentiment estimation model using SS–SVR for the SNS documents.

Bibliography

Almeida, M. B., Braga, A. P., Braga, J. P., 2000. SVM-KM: Speeding SVMs learning with a priori cluster selection and k-means. In: Proceedings of the 6th Brazilian Symposium on Neural Networks. Rio de Janeiro, Brazil, pp. 162–167.

Bakir, G. H., Bottou, L., Weston, J., 2005. Breaking SVM complexity with cross-training. Advances in Neural Information Processing Systems 17, 81–88.

Berry, M. J., Linoff, G. S., 1997. Data Minig Techniques. Wiley, New York, NY, USA.

Berry, M. J., Linoff, G. S., 2000. Mastering Data Mining. Wiley, New York, NY, USA.

Besnard, J., Toprac, A., 2006. Wafer-to-wafer virtual metrology applied to run-to-run control. In: Proceedings of the 3rd ISMI Symposium on Manufacturing Effectiveness. USA.

Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Springer, Cambridge, UK.

Blatberg, R. C., Kim, B.-D., Neslin, S. A., 2008. Chapter 10. The predictive modeling process. In: Database Marketing: Analyzing and Managing Customers. Springer, New York, USA, pp. 245–287.

Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with

co-training. In: COLT: Proceedings of the Workshop on Computational Learning Theory. New York, NY, USA, pp. 92–100.

Brefeld, U., Gärtner, T., Scheffer, T., Wrobel, S., 2006. Efficient co-regularised least squares regression. In: Proceedings of the 23rd International Conference on Machine Learning. Pittsburgh, PA, USA.

Breiman, L., 1996. Bagging predictors. Machine Learning 24 (2), 123–140.

Burges, C. J., 1998. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 121–167.

Chang, Y., Kang, Y., Hsu, C., Chang, C., Chan, T., 2006. Virtual metrology technique for semiconductor manufacturing. In: Proceedings of the 2006 International Joint Conference on Neural Networks. Vancouver, Canada, pp. 5289–5293.

Chapelle, O., Schölkopf, B., Zien, A., 2006. Semi-Supervised Learning. MIT Press, Cambridge, UK.

Chawla, N. V., Japkowicz, N., Kotcz, A., 2004. Special issue on learning from imbalanced data sets. ACM SIGKDD Explorations Newsletter 6 (1), 1–6.

Chen, P., Wu, S., Lin, J., Ko, F., Lo, H., Wang, J., Yu, W., Liang, M., 2005. Virtual metrology: a solution for wafer to wafer advanced process control. In: IEEE International Symposium on Semiconductor Manufacturing. USA, pp. 155–157.

Cheng, J., Cheng, F., 2005. Application developmetn to virtual metrology in semiconductor industry. In: Proceedings of the 32nd Annual Conference of IEEE Industrial Electronics Society. USA, pp. 124–129.

Cortes, C., Mohri, M., 2006. On transductive regression. Tech. rep., New York University.

Davenport, T. H., Harris, J. G., 2007. Competing of Analytics. Harvard Business School Press, Boston, MA, USA.

de SA, V. P., 1993. Learning classification with unlabeled data. Advances in Neural Information Processing Systems (NIPS).

Déniz, O., Castrillón, M., Hernández, M., 2003. Face recognition using independent component analysis and support vector machines. Pattern Recognition Letters 24 (13), 2153–2157.

Duda, R. O., Hart, P. E., Stork, D. G., 2001. Pattern Classification. John Wiley and Sons, New York, NY, USA.

Gönül, F. F., Kim, B.-D., Shi, M., 2000. Mailing smarter to catalog customer. Journal of Interactive Marketing 14 (2), 2–6.

Guo, G., Zhang, J.-S., 2007. Reducing examples to accelerate support vector regression. Pattern Recognition Letters 28, 2173–2183.

Ha, K., Cho, S., MacLachlan, D., 2005. Response models based on bagging neural networks. Journal of Interactive Marketing 19 (1), 17–30.

Haughton, D., Oulabi, S., 1997. Direct marketing modeling with cart and chaid. Journal of Interactive Marketing 11 (4), 42–52.

Haykin, S., 1999. Neural Networks. Prentice Hall, Upper Saddle River, NJ, USA.

Hosmer, D. W., Lemeshow, S., 2000. Applied Logistic Regression. John-John Wiley and Sons, New York, NY, USA.

Joachims, T., 1998. Text categorization with support vector machines: Learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning. Chemnitz, Germany, pp. 137–142.

Joachims, T., 1999. Transductive inference for text classification using support vector machines. In: Proceedings of the 16th International Conference on Machine Learning. pp. 202–209.

Joachims, T., 2006. Training linear SVMs in linear time. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Philadelphia, USA, pp. 217–226.

Johnson, R. A., Wichern, D. W., 1998. Applied Multivariate Statistical Analysis. Prentice Hall, Upper Saddle River, NJ, USA.

Kang, P., 2010. Ph.D Thesis: Locally Linear Reconstruction for Prediction, Detection and Clustering. Seoul National University, Seoul, Korea.

Kang, P., Cho, S., 2008. Locally linear reconstuction for instance-based learning. Pattern Recognition 41, 3507–3518.

Kang, P., Kim, D., Lee, H.-J., Doh, S., Cho, S., 2011. Virtual metrology for run-to-run control in semiconductor manufacturing. Expert Systems with Applications 38 (3), 2508–2522.

Kang, P., Lee, H.-J., Cho, S., Kim, D., Park, J., Park, C., Doh, S., 2009. A virtual metrology system for semiconductor manufacturing. Expert Systems with Applications 36 (10), 12554–12561.

Kim, D., Cho, S., 2006. e–tube based pattern selection for support vector machines. Lecture Notes in Computer Science (LNCS) 3918, 215–224.

Kim, D., Cho, S., 2008. Bootstrap based pattern selection for support vector regression. Lecture Notes in Computer Science (LNCS) 5012, 608–615.

Kim, D., Cho, S., 2010. A hybrid customer score for response modeling using support vector regression with pattern selection. In: Presented in 2010 Institute for Operations Research and the Management Sceinces (INFORMS) Annual Meeting. Austin, TX, USA.

Kim, D., Cho, S., 2012. Pattern selection for support vector regression based response modeling. Expert Systems with Applications 39 (10), 8975–8985.

Kim, D., Kang, P., Cho, S., Lee, H.-J., Doh, S., 2012. Machine learning-based novelty detection for faulty wafer detection in semiconductor manufacturing. Expert Systems with Applications 39 (4), 4075–4083.

Kim, D., Lee, H.-J., Cho, S., 2008. Response modeling with support vector regression. Expert Systems with Applications 34 (2), 1102–1108.

Kourti, T., MacGregor, J. F., 1995. Process analysis, monitoring and diagnosis, using multivariate projection methods. Chemometrics and Intelligent Laboratory Systems 28 (1), 3–21.

Lee, H.-J., Cho, S., 2007. Focusing on non-respondents: Response modeling with novelty detectors. Expert Systems with Applications 33 (2), 522–530.

Lee, H.-J., Shin, H., Hwang, S.-S., Cho, S., MacLachlan, D., 2010. Semi-supervised response modeling. Journal of Interactive Marketing 24 (1), 42–54.

Lee, S.-K., Kang, P., Cho, S., 2012a. Probabilistic local reconstruction in k-NN regression problem. In: Proceedings of the 2012 The Korean Operations Research and Management Science Society Conference. Seoul, Korea.

Lee, S.-K., Kang, P., Cho, S., 2012b. Probabilistic local reconstruction for k-NN regression. Neurocomputing, Submitted.

Li, H., Liang, Y., Xu, Q., 2009. Support vector machines and its applications in chemistry. Chemometrics and Intelligent Laboratory Systems 95 (2), 188–198.

Malthouse, E. C., 1999. Ridge regression and direct marketing scoring models. Journal of Interactive Marketing 13 (4), 10–23.

Malthouse, E. C., 2002. Performance–based variable selection for scoring models. Journal of Interactive Marketing 16 (4), 37–50.

Maulik, U., Chakraborty, D., 2011. A self-trained ensemble with semisupervised SVM: An application to pixel classification of remote sensing imagery. Pattern Recognition 44 (3), 615–623.

Mitchell, T. M., 1996. An introduction to genetic algorithms. MIT Press, Cambridge, UK.

Mitchell, T. M., 1997. Machine Learning. McGraw-Hill, Singapore.

Mitchell, T. M., 1999. The role of unlabeled data in supervised learning. In: Proceedings of the 6th International Colloquium on Cognitive Science. San Sebastian, Spain.

Ortiz-García, E., Salcedo-Sanz, S., Pérez-Bellido, A., Portilla-Giqueras, J., Prieto, L., 2010. Prediction of hourly o3 concentrations using support vector regression algorithms. Atmospheric Environment 44, 4481–4488.

Pai, P.-F., Lin, C.-S., 2005. A hybrid arima and support vector machines model in stock price forecasting. Omega 39 (6), 497–505.

Platt, J. C., 1998. Fast training of support vector machines using sequential minimal optimization. In: Advances in Kernel Methods - Support Vector Learning. MIT Press, Cambridge, MA, USA, pp. 41–65.

Qin, S. J., 2003. Statistical process monitoring: basics and beyond, journal of chemometrics. Journal of Chemometrics 17, 480–502.

Qin, S. J., Cherry, G., Good, R., Wang, J., Harrison, C. A., 2006. Semiconductor manufacturing process control and monitoring: A fab-wide framework. Journal of Process Control 16 (3), 179–191.

Schölkopf, B., Smola, A., 2002. Learning with Kernels. MIT Press, Cambridge, UK.

Sen, A., Srivastava, M., 1990. Regression Analysis: Theory, Methods, and Applications. Springer-Verlag, New York, NY, USA.

Shalev-Shwartz, S., Singer, Y., Srebro, N., 2007. Pegasos: Primal estimated sub-gradient solver for svm. In: Proceedings of the 24th Interational Conference on Machine Learning. Corvallis, OR, USA, pp. 807–814.

Shawe-Taylor, J., Cristianini, N., 2000. An Introduction to Support Vector Machines. Cambridge University Press, Cambridge, UK.

Shin, H., 2005. Ph.D Thesis: Efficient Pattern Selection for Support Vector Classifiers and its CRM Application. Seoul National University, Seoul, Korea.

Shin, H., Cho, S., 2006. Response modeling with support vector machines. Expert Systems with Applications 30 (4), 746–760.

Shin, H., Cho, S., 2007. Neighborhood property based pattern selection for SVM. Neural Computation 19 (3), 816–855.

Shmueli, G., Patel, N. R., Bruce, P. C., 2007. Data Mining for Business Intelligence. Wiley, New York, NY, USA.

Sindhwani, V., Niyogi, P., Belkin, M., 2005. A co-regularized approach to semi-supervised learning with multiple views. In: Proceedings of the 22nd ICML Workshop on Learning with Multiple Views. Bonn, Germany, pp. 217–226.

Smola, A., Schölkopf, B., 2002. A tutorial on support vector regression. Tech. Rep. NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, UK.

Su, A.-J., Jeng, J.-C., Huang, H.-P., Yu, C.-C., Hung, S.-Y., Chao, C.-K., 2007. Control relevant issues in semiconductor manufacturing: Overview with some new results. Control Engineering Practice 15 (10), 1268–1279.

Sun, A., Liu, Y., Lim, E.-P., 2011. Web classification of conceptual entities using co-training. Expert Systems with Applications 38 (12), 14367–14375.

Sun, J., Cho, S., 2006. Pattern selection for support vector regression based on sparsity and variability. In: Procedings of 2006 IEEE International Joint Conference on Neural Networks (IJCNN). Vancouver, BC, Canada, pp. 559–602.

Thissen, U., van Brakel, R., de Weijer, A., Melssen, W., Buydens, L., 2003. Using support vector machines for time series prediction. Chemometrics and Intelligent Laboratory Systems 69 (1-2), 35–49.

Vapnik, V., 1995. The Natural of Statistical Learning Theory. Springer, New York, USA.

Wang, K., Zhou, S., Yang, Q., Yeung, J. M. S., 2005. Mining customer value: from association rules to direct marketing. Data Mining and Knowledge Discovery 11, 57–79.

Wang, W., Xu, Z., 2004. A heuristic training for support vector regression. Neurocomputing 61, 259–275.

Wang, X., Fu, L., , Ma, L., 2011. Semi-supervised support vector regression model for remote sensing water quality retrieving. Chinese Geographical Science 21 (1), 57–64.

Wang, X., Ma, L., Wang, X., 2010. Apply semi-supervised support vector regression for remote sensing water quality retrieving. In: Procedings of 2010 IEEE International Geoscience and Remote Sensing Symposium. Honolulu, HW, USA, pp. 2757–2760.

Widodo, A., Yang, B.-S., Han, T., 2007. Combination of independent component analysis and support vector machines for intelligent fault diagnosis of induction motors. Expert Systems with Applications 32 (2), 299–312.

Yang, J., Honavar, V., 1998. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems 13 (2), 44–49.

Yu, E., Cho, S., 2006. Constructing response model using ensemble based on feature subset selection. Expert Systems with Applications 30 (2), 352–360.

Zhou, Z.-H., Li, M., 2007. Semisupervised regression with cotraining-style algorithms. IEEE Transactions on Knowledge and Data Engineering 19 (11), 1479–1493.

Zhu, X., 2006. Semi-supervised learning literature survey. Tech. Rep. 1350, Department of Computer Science, University of Wisconsin at Madison, Madison, WI, USA.

초 록

Support Vector Regression(SVR) 알고리즘은 Support Vector Machines(SVM)의 회귀분석 버전이다. SVR 은 SVM 과 마찬가지로 구조적 위험 최소화(Structural Risk Minimization: SRM) 원리를 통해 일반화 성능(generalization performance)를 높일 수 있으며, 커널(kernel) 함수를 이용하여 비선형 문제를 해결할 수 있다. 위와 같은 장점을 바탕으로 SVR 은 현재 가장 각광받고 있는 회귀분석 모델이다. 그러나 SVR 에서도 아직 해결되지 못한 문제들이 남아있다. 본 논문에서는 가장 큰 두 가지 문제를 다루고자 한다.

첫 번째는 학습 복잡도(training complexity) 문제이다. SVR 의 학습 복잡도는 대용량의 데이터를 학습하기에 너무 크다. SVR 의 학습 복잡도는 학습 데이터의 수와 밀접한 관련이 있는데, 학습 시간 복잡도는 $O(n^3)$이고 학습 메모리 복잡도는 $O(n^2)$이다(n 은 학습 데이터의 수). 본 논문에서는 학습 복잡도를 줄이기 위해 Margin based Data Selection(MDS) 알고리즘을 제안한다. SVR 의 학습 복잡도는 학습 데이터의 수에 가장 큰 영향을 받기 때문에, 학습 복잡도를 줄이기 위해선 학습 데이터의 수를 줄이는 것이 가장 효과적인 방법이다. 데이터 선택(data selection) 기법은 학습에 유용한 소수의 데이터를 전체 학습 데이터로부터 선택하여 새로운 학습 데이터 셋을 만드는 것을 의미한다. SVR 학습에서 가장 중요한 데이터는 Support Vectors(SVs)이다. 단, SVR 의 SVs 는 ε-loss 방식에 따라 ε-tube 의 경계 위 혹은 바깥쪽에 위치하게 된다. MDS 는 다중 샘플 학습을 통해 학습 데이터의 평균 마진(averaged margin)을 계산한 후, 평균 마진과 ε과의 비교를 통해 SVs 가 될 가능성이 높은 데이터를 선택한다. MDS 의 성능 평가를 위해, 20 개의 데이터 셋을 이용한 실험을 진행하였다. 그 결과, DMS 는 다른 비교 방법들보다 우위의 성능을 보여주었다. 전체 데이터를 학습할 때와 비교해서 평균 38~67%의 학습 시간만을 이용하고도 학습이 완료되었다. 그와 동시에 평균적인 정확도는 0~1%의 저하만 있었다.

두 번째는 반교사 학습(semi-supervised learning) 문제이다. 일반적인 교사 학습(supervised learning)은 타겟 값이 있는 labeled 데이터만을 사용하여 학습한다. 하지만 실제 데이터의 대부분은 타겟 값이 존재하지 않는 unlabeled 상태로 존재한다. 반교사 학습은 이러한 unlabeled 데이터를 labeled 데이터와 함께 학습하여 모델의 성능을 높이려는 목적으로 제안되었다. 분류 문제와는 달리 회귀 문제의 경우 예측해야 하는 unlabeled 데이터의 타겟이 연속형 변수이기 때문에 더 어려움이 있다. 본 논문에서는 SVR 의 반교사 학습 모델(SS-SVR)을 제안하였다. Unlabeled 데이터의 타겟을 예측할 때, 레이블 값이 아닌 레이블 분포를 예측하기 위해 확률 기반 지역 재구축 방법인 Probabilistic Local Roconstruction(PLR)을 사용하였다. 단, PLR 은 노이즈 데이터에 민감할 수 있으므로 두 개의 PLR 모델을 사용하여, 각각 지역적 재구축과

전역적 재구축을 담당하게 하였다. 그리고 두 모델의 결과를 다시 조합(conjugation)하여 최종적인 unlabeled 데이터의 레이블 분포를 예측하였다. 그 후, 각 unlabeled 데이터의 불확실성(uncertainty)에 따라 학습 데이터를 생성(generation)하였다. 학습 데이터는 unlabeled 데이터와 그 레이블 분포에 따라 여러 개를 생성하는데, 불확실성이 높은 데이터는 정보량을 늘려주기 위해 다수를 생성하고, 불확실성이 낮은 데이터는 중복을 막기 위해 소수를 생성하였다. 그 후 기존에 제안했던 MDS 기법을 사용하여 학습 데이터의 수를 적정 수준으로 유지하여서 학습 복잡도를 감소시켰다. 제안 SS-SVR 기법의 성능 평가를 위해, 16 개의 데이터 셋을 이용한 실험을 진행하였다. 제안 기법은 labeled 데이터만 학습하는 교사 학습 방법의 SVR 에 비해 평균 10% 정도의 정확도 향상을 기록하였다. 이 성능은 비교 방법들 중에서도 우수한 성능에 해당된다. 또한 제안 기법은 비교 방법들 중 가장 단축된 학습 시간을 보여주었다. 비교 방법들 중 가장 빠른 방법 대비 평균 25%의 학습 시간만으로 비슷한 성능의 학습을 진행할 수 있었다.

제안된 두 가지 방법론의 실제 문제로의 적용성을 평가하기 위해, 실제 데이터셋에 적용하는 실험을 하였다. 첫 번째로 반응 모델링(response modeling)에 MDS 를 적용하였다. 이 과정에서 기존 반응 모델링과는 달리 분류 모델과 회귀 모델을 함께 사용하는 2 단계 반응 모델(two-stage response model)을 제안하였다. 성능 평가는 반응 모델의 수익률로 평가하였는데, 2 단계 반응 모델은 기존 반응 모델보다 더 높은 수익률을 보여주었다. 동시에 MDS 를 적용한 2 단계 반응 모델은 최소한의 수익률 저하와 함께 SVR 의 학습 복잡도를 60% 수준으로 감소시킬 수 있었다. 두 번째로는 가상 계측(virtual metrology)에 SS-SVR 을 적용하였다. 실제 반도체 공정에서 얻어진 가상 계측 데이터셋에 SS-SVR 을 적용한 결과, 평균적으로 8%의 성능 향상이 이루어졌다. 이는 비교 방법들 대비 우수한 성능이었다. 또한 학습 시간 역시 비교 방법들 중 가장 단축된 시간을 기록하였다.

주요어: 데이터마이닝(data mining), 패턴 인식(pattern recognition), Support Vector Machines(SVM), Support Vector Regression(SVR), 반교사 학습(semi-supervised learning), 데이터 선택(data selection), 데이터 생성(data generation), 고객 관계 관리(customer relationship management), 반응 모델링(response modeling), 반도체 공정(semiconductor manufacturing), 가상 계측(virtual metrology)

학 번: 2005-20821