공학박사 학위논문

Sparse Learning Models and Their Applications to Financial Technologies

축약 학습 방법론과 금융 기술 문제에의 적용

2015 년 8 월

서울대학교 대학원

산업조선공학부 산업공학 전공

손 영 두

Sparse Learning Models and Their Applications to Financial Technologies

축약 학습 방법론과 금융 기술 문제에의 적용

지도교수 이 재 욱

이 논문을 공학박사 학위논문으로 제출함

2015 년 6 월

서울대학교 대학원

산업조선공학부

손 영 두

손영두의 공학박사 학위논문을 인준함

2015 년 6 월

위 원 장	조 성 준	(인)
부위원장	이 재 욱	(인)
위 원	장 우 진	(인)
위 원	김 성 범	(인)
위 원	이 종 석	(인)

학위논문 원문 이용에 대한 동의서

본인은 아래의 학위논문이 제3자의 권리를 침해하지 않았음을 서약하며, 서울대학교가 다음과 같이 저작물을 이용하는 것에 동의합니다.

논문 제목	Sparse Learning Models and Their Applications to Financial Technologies
학위 구분	석사 □ / 박사 ☑
학 과	산업조선공학부
학 번	2012-30287
연 락 처	010-7206-0419

1. 본인은 서울대학교가 위 저작물을 인터넷 등 정보통신망을 통해 복제·전송·배포하는 것에 동의합니다.

2. 본인은 서울대학교가 위 저작물에 대해 무료로 온라인 서비스를 제공하는 것에 동의합니다.

3. 서울대학교는 내용을 변경하지 않는 범위 안에서 위 저작물을 다른 파일 형식으로 변경할 수 있습니다.

4. 본인은 위 저작물의 저작권을 타인에게 양도하거나 출판을 허락하는 등 동의 내용을 변경하고자 할 경우 소속대학(원)에 공개의 유보 또는 해지를 즉시 통보하겠습니다.

5. 서울대학교는 저작권법 및 도서관법을 준수하며 해당 저작물로 인하여 발생하는 타인에 의한 권리 침해에 대하여 일체의 법적 책임을 지지 않습니다.

제 출 일 : 2015 년 6 월 15 일

저 작 자 : 손 영 두 (인)

서울대학교총장 귀하

Abstract

Sparse Learning Models and Their Applications to Financial Technologies

Youngdoo Son
Department of Industrial Engineering
and Naval Architecture
The Graduate School
Seoul National University

As an era of big-data arises, the more efficient algorithms, in the sense of both time and storage, are required for data analysis. The sparse learning models satisfy these requirement, maintaining the ability of existing learning models to describe data distribution well. Therefore, the sparse learning models have been studied enormously from the middle of 2000s. Also, as developed data storage techniques have been applied to several business area, including finance, these sparse models obtain some possibilities to construct an accurate and efficient model compared to the existing parametric models.

In this dissertation, we developed two novel sparse learning models using a kernel method and the automatic relevance determination prior. Then, several learning models, including both sparse and non-sparse ones, are applied to two financial applications related to the financial technology,

The first developed model is a sparse support-based clustering model with a support function derived from the variance function of Gaussian process (GP) regression using automatic relevance determination prior and variable GP noise to overcome these clustering problems. The proposed method has a distinct feature that the support function is represented by a smaller number of representative vectors (center of kernels) than those of in previous studies. Another feature of the proposed method is that these representative vectors are in the training data set and are automatically located during the training process. Simulation result for various clustering problems show that the proposed method significantly reduces the labeling time. The exemplars of handwritten digit data sets selected using the proposed method are also reported.

The second model is an active learning algorithm for sparse Bayesian regression. Active learning is one of large and important branches in machine learning and it aims to build an accurate learning model with a relatively small number of labeled points which are chosen actively by the constructed learning model. Active learning algorithms are usually required when the cost of gaining labels of data points is expensive. We propose two sub-steps to construct the proposed algorithm. First, we develop a transductive and generalized version of relevance vector machine which obtains its basis vectors from the unlabeled data set as well as the labeled one. Next, we suggest three querying strategies which uses only the relevance vectors automatically selected by the developed model for active selection for data points to be labeled. The proposed method were applied to several artificial and real data sets and showed better performance than the benchmark, random selections, and these results were statistically significant in

most cases.

As learning model applications to financial data, we pay attention to the predictions of two financial variables: the market impact costs and the credit default swap spreads. The first variable, the market impact cost, have not been analyzed by machine learning algorithms before and the learning application for the second variable has been rarely studied, but none of them applied several state-of-the-art learning models and compared the results among them.

For the prediction task of market impact cost, we applied two sparse learning models, support vector regression and relevance vector machine, and three non-sparse models, neural networks, Bayesian neural networks, and Gaussian process, to single transaction data of US equity market and compared their performances with one another and the benchmark parametric model. The active learning algorithm developed in chapter 4 was also applied to predict the market impact cost. As a result, the learning models except the support vector regression showed better performance than the parametric benchmark and the active learning algorithm performed better than the random selection with much lower number of labeled points than the full sparse Bayesian regression model.

For the prediction task of credit default swap spreads, we applied the same five learning models and also a parametric benchmark to daily credit default swap spreads from 2001 to 2014, which includes the global financial crisis period when the credit risk of firms were very high, and compared their performances with one another. Also in this application, support vector regression caused bad results especially when the credit risk is high. The relevance vector machines

showed much better performances than the support vector regression but worse than the other non-sparse learning models.

Keywords: clustering, active learning, sparse Bayesian, financial technology

Student Number: 2012-30287

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, the motivations and aims of the research topics in this dissertation are given with the overviews of the topics. Then, the organization of the remaining chapters of the dissertation is provided.

1.1 Motivation of the Dissertation

With a rapid development of storage and digitalization of data, the era of big data arises. Huge amount of data can be stored and accessible in various areas including meteorology, genomics, physics, mechanical and electrical engineering, business, and finance. Analysis of those large scale data with the traditional non-sparse models requires both time and storage a lot. Therefore, more efficient techniques for data analysis are required and become more important. The sparse learning models, one of those efficient techniques, are explored again in recent days.

Among sparse models, sparse kernel machines have both advantages of kernel methods and sparse models: detect and fit well the complex data distribution and represented by only a few basis vectors. One of the famous sparse kernel

machines is the relevance vector machine (RVM) which employs the automatic relevance determination (ARD) prior distribution for the its Bayesian nature. This ARD prior makes the model very sparse, even sparser than the famous support vector machine (SVM)(Boser et al., 1992; Vapnik, 2000).

There have been a lot of machine learning approaches for financial problems from a few decades ago(Hutchinson et al., 1994; S. H. Kim & Noh, 1997; W.-H. Chen et al., 2006; Ticknor, 2013) leaning on its old tradition of collecting data for model verification and financial modeling. With the improvement storage and digitalization skills for data, financial data also rapidly increases in both volume and variety. Also, due to newly developed area of financial technology, data analysis and learning models are required in financial fields other than ones that the learning algorithms have already been applied to.

Hence, we paid a lot of intention to develop the sparse models using ARD prior and apply state-of-the-art learning models including sparse ones to the tasks of estimating and predicting financial variables that the learning models have yet barely been applied to. In this dissertation, two novel learning algorithms using ARD prior are proposed and two financial application examples of learning models are presented.

1.2 Aims of the Dissertation

The aim of this dissertations is to develop novel sparse learning algorithms that can aid big data analyses and to apply several state-of-the-art learning models to improve predicting performances of financial variables to examine

their applicability to the financial technologies. The topics to be considered in this dissertation include clustering, active learning, and applications to financial technology problems. For the first two topics, the sparse learning algorithms are developed to improvement performances and sparsities of them compared to the existing or benchmark model. For the last topic, several machine learning models, including both sparse and non-sparse ones, are applied to two financial problems related to the financial technology, estimating transaction costs and credit risks. The detailed research objective of each problem is as follows:

- **Developing Sparse Support-based Clustering using Automatic Relevance Determination (Chapter 3)**: Support-based clustering methods, such as support vector clustering and Gaussian process clustering (GPC), despite of their ability to represent clusters with complex shapes, suffer from expensive computational cost in the training-labeling stage and test-clustering phase for large-scale nonconvex clustering problems. In chapter 3, we propose a novel sparse support-based clustering method with a support function derived from the variance function of Gaussian process (GP) regression using ARD prior and variable GP noise to overcome these clustering problems. One distinct feature of the proposed method is that its support function is represented by a smaller number of representative vectors (center of kernels) than those of in previous studies. Another feature is that these representative vectors are in the training data set and are automatically located during the training process. The proposed method is applied to the various clustering data sets to examine

its operability and characteristics.

- **Developing Active Learning Method for Transductive Sparse Bayesian Regression (Chapter 4)**: As one of the most important and practical areas in machine learning and data mining, active learning aims to build an accurate learning model with a relatively small number of labeled points that are chosen actively by the constructed learning model. Active learning algorithms play an important role in knowledge-based systems when the cost of obtaining labeled data points is expensive. In chapter 4, we propose an active learning algorithm for transductive sparse Bayesian regression. First, we develop a transductive and generalized version of the RVM, which obtains its basis vectors from the unlabeled data set as well as the labeled one. Then, we suggest three querying strategies for active learning, which only use the relevance vectors automatically selected by the developed model for active selection for data points to be labeled.

- **Applying Learning Models to Problems Related to Financial Technologies (Chapter 5)**: From a few decades ago, several recently developed technologies have been applied to financial markets. Learning models have also been employed to estimate and predict financial variables and market parameters and resulted in better performances than the in several financial markets. However, there are still remaining the financial problems the learning models have not been applied yet but may have possibilities to help to solve those problems. In chapter 5, we pay attention

to the predictions of two financial variables: the market impact costs and the credit default swap spreads. For the prediction task of market impact cost, we applied two sparse learning models, support vector regression (SVR) and RVM, and three non-sparse models, neural networks (NNs), Bayesian neural networks (BNNs), and GP, to single transaction data of US equity market and compared their performances with one another and the parametric benchmark, I-star model(Kissell et al., 2003; Kissell, 2013). The active learning algorithm developed in chapter 4 was also applied to predict the market impact cost. For the prediction task of credit default swap spreads, we applied the same five learning models and also one parametric benchmark model, the constant intensity model(Jarrow & Turnbull, 1995), to daily credit default swap spreads from 2001 to 2014 and compared their performances with one another.

1.3 Organization of the Dissertation

The remainder of this dissertation is organized as follows. In the next chapter, we review history and previous results of sparse learning models, active learning, and financial applications of learning models. Then, in chapter 3, we propose a novel sparse support-based clustering using ARD, which is much sparser and reduces the labeling time compared to the previous models. In chapter 4, we very firstly propose the active learning model for RVM as developing a transductive and generalized version of RVM and suggest three querying strategies to actively select data points to be labeled. We applied the learning models, both sparse

and non-sparse ones, to the prediction tasks of the market impact costs and credit default swap spreads and compared the performances with one another and the parametric benchmarks as well in chapter 5. Finally, in chapter 6, we conclude this dissertation with summary and possible future work of the research.

Chapter 2

Literature Review

Since Arthur Samuel first defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed"(Simon, 2013) and the first perceptron was developed(Rosenblatt, 1961), the machine learning models have received enormous attention from both academia and industry. In the middle of 1980s, the main theme of machine learning was the multilayer NNs. Rumelhart et al. (1985) proposed the back-propagation algorithm by which the weights of the multilayer NN model could be trained efficiently and Cybenko (1989) studied the properties of activation functions which are employed to represent the nonlinear distributions of data points. Support vector algorithms were an avalanche of mid-1990s and early 2000s. The first SVM was proposed in Boser et al. (1992) and it became able to deal with the non-separable cases by including the slack variables(Cortes & Vapnik, 1995). It was also extended to the regression model by Drucker et al. (1997). Two previously-mentioned algorithms, NN models and SVMs, has been successfully adapted to diverse fields in natural sciences, social sciences, engineering, and real business applications.

As the storage technologies of data has been improved after the middle of

2000s, the research on machine learning models proceeds in two ways. The first approach is to lean on the development of computing power. The research of this approach focuses on modifying algorithm for using the improved computing methods including parallel computing and GPU computing. Solving the problems that requires huge computation, like deep learning, is also a huge branch of this theme. The second approach concentrates on modifying the model efficiently to handle the large scale data sets. The sparse models are typical research topics of this approach.

In this dissertation, as mentioned in the previous chapter, two novel sparse model using ARD prior distribution are proposed and two financial application of the state-of-the-art machine learning models including both sparse and non-sparse ones are presented. We briefly review the related literatures in which these topics of this dissertation are embedded in the following sections.

2.1 Sparse Learning Models

The research on sparse models has been conducted in different directions. In this section, we review the history of two different types of research on sparse models: sparse linear models and sparse kernel models.

2.1.1 Sparse linear models

Sparse linear model algorithms are based on the generalized linear model

$$p(y|\mathbf{x}) = p(y|f(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}))) \tag{2.1}$$

where $\boldsymbol{\phi}$ is a basis vector and f is a link function. The aim of sparse linear model is to represent the model with the smaller number of nonzero entries of the weight vector $\mathbf{w}$. This is achieved by giving a regularization on it. ℓ_0-norm and ℓ_1-norm regularizations are the most typical approaches for those regularizations.

The most famous ℓ_1-norm regularization method is the least absolute shrinkage and selection operator, usually known as *LASSO*, proposed in Tibshirani (1996). With the predicted output $\hat{y} = f(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}))$, the LASSO regression aims to find the weight vector $\mathbf{w}$ satisfying

$$\min_{\mathbf{w}} \mathcal{L}(y, \hat{y}) + \lambda \|\mathbf{w}\|_1 \tag{2.2}$$

where $\mathcal{L}$ is a loss function between two outputs and λ is a parameter for the regularization. Theoretically, this type of regularization is equivalent to giving Laplace prior distribution of the weight vector, whose mean is 0 and scale factor is $1/\lambda$. There have been suggested several methods to solve this optimization problem efficiently.

W. J. Fu (1998) and T. T. Wu and Lange (2008) used the coordinate descent method to finding an optimal solution. Efron et al. (2004) suggested the least angle regression and shrinkage method, *LARS*, which provides the curve denoting the solution for each value of the regularization parameter λ. Wright et al. (2009) and Nesterov (2004) used proximal operators and Figueiredo (2003) used expectation-maximization method to optimize the regularized regression problem in (2.2). Yuan and Lin (2006) extended LASSO regression to the group LASSO regression, in which the sparsity is applied to the groups of input fea-

tures.

Although ℓ_0-norm regularization causes the regression model sparser than ℓ_1-norm regularization, finding optimal solution for (2.2) becomes much difficult if the regularization term of the weight vector is changed to ℓ_0-norm. Therefore, the optimization procedure for ℓ_0-norm regularization have been developed for relatively recent years. Soussen et al. (2011) and S. Chen and Wigger (1995) used greedy search to find the optimal weights and O'Hara et al. (2009) and Bottolo et al. (2010) employed the stochastic approaches. In addition, J. C. Huang et al. (2007) and Rattray et al. (2009) suggested variational inference methods for this optimization problem.

2.1.2 Sparse kernel models

From the development of SVMs(Boser et al., 1992), sparse kernel models are another huge branch of sparse learning algorithms. We review some extensively used sparse kernel machines here.

SVMs(Boser et al., 1992) are the sparse kernel classification method which finds the optimal hyperplane, which maximizes the margin of the classifier. Vapnik (2000) allowed the misclassification as giving them a penalty term so SVMs became able to be adapted to the non separable case. Drucker et al. (1997) extended the SVMs to regression problem using ϵ-insensitive loss function, i.e. $\mathcal{L}(y_1, y_2) = \max\{\epsilon, |y_1 - y_2|\} - \epsilon$.

There exist ℓ_1-regularized vector machines(Krishnapuram et al., 2005). ℓ_1-regularized vector machines use kernel functions as basis functions like other usual kernel regression methods but the weights for these basis functions are

regularized with ℓ_1-norm.

RVMs(Tipping, 2001) is another famous sparse kernel machine method. In RVMs, the weight vector of the basis kernel functions has the ARD prior, i.e., $p(\mathbf{w}|\mathbf{A}) \sim \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{A}^{-1}) = \prod_{i=1}^{N} \mathcal{N}(w_i|0, A_{ii}^{-1})$. The most important feature of RVMs is that it results in the sparser model than the SVMs and the ℓ_1-regularized vector machines. Tipping and Faul (2003) suggested the fast marginalization method to train the RVM faster and Wipf and Nagarajan (2008) provided the theoretical background of the sparseness of the ARD prior in the view of traditional optimization.

There have been some trials to introduce the sparsity to GPs(Cressie, 1993; Rasmussen, 1996). For example, Snelson and Ghahramani (2006) made pseudo inputs and Csató and Opper (2002) used the online approach to construct the sparse version of GPs.

Relative to supervised learning, there have been lack of attention to unsupervised and semi-supervised version of sparse learning. However, there also exist some milestones that should be reviewed.

Tax and Duin (1999) first suggested the methods for finding support functions represented by the small number of support vectors and able to capture the complex shape of data distributions, there have been a lot of studies to cluster the points with this support function. These clustering algorithm is called *support vector clustering*. Since finding the cluster labels of data points requires high computational costs, many works have been devoted to improve the speed of the labeling step since the complete graph labeling method was proposed in Ben-Hur et al. (2002). Examples of these works include the approximated

graph techniques (J. Yang et al., 2002), spectral graph partitioning strategy (J. Park et al., 2004), ensembles combined strategy (Puma-Villanueva et al., 2005), chunking strategies (Ban & Abe, 2004), pseudohierarchical technique (Hansen et al., 2007), equilibrium vector-based clustering techniques (J. Lee & Lee, 2005, 2006), fast support vector clustering (Jung et al., 2010), cone cluster labeling (S.-H. Lee & Daniels, 2012), and Vornoi cell-based approach(K. Kim et al., 2015). Additionally, Tsang et al. (2005) suggested another way of finding the support function requiring much lower time and space complexities using approximation techniques.

For semi-supervised version of sparse kernel machines, there exists some results for the semi-supervised classification of SVM. Joachims (1999) proposed the transductive SVMs which give uncertain outputs to unlabeled data points and find optimal hyperplane and optimal labels for unlabeled points simultaneously. D. Lee and Lee (2007) suggested the semi-supervised classfication technique based on the clusters resulted from support vector clustering.

2.2 Active Learning

As one of the most important and practical areas in machine learning and data mining, active learning aims to build an accurate learning model with a relatively small number of labeled points that are chosen actively by the constructed learning model.

Most studies on active learning have been focused on establishing the efficient querying strategy. One of the simplest querying strategies is uncertainty

sampling (Lewis & Catlett, 1994). Basic uncertainty sampling including margin sampling (Scheffer et al., 2001) queries the most uncertain point in the classification case, the point that has the maximum probability belonging to each class is the smallest, and in the regression case, the point that has the highest variance among the unlabeled data points. This strategy, which is based on querying the most uncertain points among the unlabeled data pool, has been successfully applied to diverse applications including image retrieval (Tong & Chang, 2001), text classification (Tong & Koller, 2002), drug discovery process (Warmuth et al., 2003), object detection problem (Vijayanarasimhan & Grauman, 2014), especially for SVM classification, as well as for other kernel classification methods such as the active kernel logistic regression (Hoi et al., 2009) that queries the labels of the multiple number of unlabeled data points and the GPs model (Kapoor et al., 2007).

Another querying strategy is to minimize the total expected variance of the model. Cohn (1996) suggested the uses of this strategy for NN models and Cohn et al. (1996) for the mixture of Gaussian regression and locally weighted regression. Since reducing the total variance of the model is equivalent to maximizing its Fisher information, Zha et al. (2012) proposed the active learning SVMs that maximize the information measure (D-optimality) and successfully applied it to video indexing application. This strategy of maximizing Fisher information is the most studied one, particularly in the field of statistics (Chaloner & Verdinelli, 1995; Flaherty et al., 2005; Schein & Ungar, 2007; Settles & Craven, 2008; Zhang & Oles, 2000).

Another widely used querying strategy is the query-by-committee strategy(Abe

& Mamitsuka, 1998; McCallum & Nigamy, 1998; Seung et al., 1992), which selects the most disagreed instance among the several committee models trained by the current labeled data set. The regression version of this strategy, as suggested by Burbidge et al. (2007), selects the unlabeled points with the largest variance among the predictions by the committee models. Other strategies, such as selecting the unlabeled point that is expected to change the model most(Settles & Craven, 2008) and to minimize the expected future error of the model(Y. Guo & Greiner, 2007; Moskovitch et al., 2007; Roy & McCallum, 2001; Zhu et al., 2003), have also been proposed recently. For further literatures on active learning, see Settles (2010) and Y. Fu et al. (2013).

2.3 Financial Applications of Learning Models

The financial variable prediction has been a long and yet active research theme targeted by many researchers since successful prediction helps to make profits as well as avoid risks. From a long time ago, many people, called chartist, have believed that the future value of financial time series can be predicted by using the past values. According to the well-known efficient market hypothesis it is argued that the stock price is fully random walk without new unpredictable information, making it almost impossible to predict it. There are, however, several counter-evidences that the stock price process does not follow the random walk leaving aside some controversial issues. Two typical such counter-evidences are the momentum effect and the mean reversion which show that the autocorrelations of the return of a stock are positive in short horizons and negative for

long horizons.

Inspired by these empirical findings, during the last decades many statistical learning technologies have been applied to predict various financial variables. The most intensively studied variable is the stock price(W.-H. Chen et al., 2006; Son et al., 2012; Ticknor, 2013; Liao & Chou, 2013) and its derivative markets(Hutchinson et al., 1994; Han & Lee, 2008; S.-H. Yang & Lee, 2011; H. Park & Lee, 2012; H. Park et al., 2014) and their predictive power were reliable in usual. There also exist learning approaches for other financial markets including fixed-income market(S. H. Kim & Noh, 1997; Cao & Tay, 2003) and foreign exchange market (Bhattacharyya et al., 2002).

Financial technology, usually known as Fintech, In a broad sense, the financial technology includes the efficient procedure for financial transactions, peer-to-peer lending, and constructing online finance systems. The machine learning applications related to the financial technology barely exist. There have been some studies focusing on credit risk and its derivative valuation(Y.-C. Lee, 2007; K.-j. Kim & Ahn, 2012; Z. Huang et al., 2004; Gündüz & Uhrig-Homburg, 2011) which may be useful for peer-to-peer lending business.

Chapter 3

Sparse Support-based Clustering using Automatic Relevance Determination

3.1 Chapter Overview

Clustering, which divides data objects into several similar groups, is one of the well-known and traditional topics in diverse fields including statistics, machine learning, and data science. Recently, support-based clustering methods with kernels have been extensively studied and successfully applied to solve many difficult clustering problems because of their ability to detect complicated non-convex shapes better than traditional clustering methods can (Ben-Hur et al., 2002; Ban & Abe, 2004; J. Lee & Lee, 2005; Girolami, 2002). The support-based clustering methods usually consist of two stages. The first stage involves constructing the support function that detects the cluster structure of a given data distribution and includes mainly the support functions constructed from the support vector domain description (SVDD) algorithm (Tax & Duin, 1999) or those constructed from the variance of the GP regression using squared exponential kernel with Gaussian noise (H.-C. Kim & Lee, 2007). The second

stage involves labeling the data points using the level set of the constructed support function and includes complete graph-based labeling (Ben-Hur et al., 2002), approximated graph technique (J. Yang et al., 2002), spectral graph partitioning (J. Park et al., 2004), ensemble combining (Puma-Villanueva et al., 2005), chunking strategy (Ban & Abe, 2004), pseudo-hierarchical technique (Hansen et al., 2007), dynamic system-based approaches (J. Lee & Lee, 2005, 2006; Jung et al., 2010), and cone cluster labeling (S.-H. Lee & Daniels, 2012).

Thus far, considerable research has focused on improving the labeling method because the second stage of labeling is the main bottleneck in the training phase as the size of the data set increases. However, in the test phase, the complexity of support-based clustering with kernels mainly depends on the computation time of the constructed support function, the time complexity of which is proportional to the number of center points used in the kernel representation. In many applications, having compact sparse models to process new data points rapidly even when it costs substantial computation time in the training phase is desirable. However, in practice, the support functions constructed from the SVDD algorithm involves computing the kernels centered at almost half of the training data points as the sample size increases, whereas those from the GP regression involves all of the training data points.

To solve this problem, we propose constructing a sparse representation of a support function with substantially reduced centered data points. We first use the GP regression model with a variance function that only depends on the training input data. We use ARD prior distribution to obtain a compact sparse model. ARD prior distribution leads to variance functions with larger

values in a dense area and smaller values in a sparse area, which is the opposite of the variance function observed in the traditional GP regression model (H.-C. Kim & Lee, 2007; Williams & Rasmussen, 2006). We also introduce a so-called variable GP noise, instead of traditional constant Gaussian noise, which enables us to obtain a GP regression model that is compatible with the GP classification model and allows us to obtain a likelihood function that utilizes the ARD prior to obtain a sparse model and improve the clustering performance. As a result, the proposed method constructs a sparse model of the estimated support function represented by a small number of basis vectors centered at some representative data points that function as exemplars in a grouped cluster.

This chapter is organized as follows. In Section 2, we present our proposed method to construct a sparse support function and detail the implementation strategy to determine the hyper-function and hyper-parameters. We also show that the proposed method can estimate the support of an unknown data distribution by using the generalization error bound. In Section 3, we show the experimental results that are applied to several kinds of clustering data sets. Section 4 provides some discussions and concludes this study.

3.2 Proposed Method

In this section, we first propose a method to construct a support function using the GP regression with the ARD prior and the so-called variable GP noise. Then, we derive a tractable likelihood function to determine hyper-functions related to a variable GP noise and hyper-parameters related to the ARD prior.

One key idea in deriving such a compact likelihood is that we can control the output values in the GP regression at our disposal to match the GP regression model with the approximated one-class GP classification model with only zero class. Then, we derive a generalization error bound to show that the obtained variance function can indeed estimate the support of a data distribution. Finally, we provide some implementation strategies to label the data points from the constructed support function.

3.2.1 Constructing the sparse support function using automatic relevance determination

First let us consider that for a pair of input-output $(\mathbf{x}, y)$, the following additive error regression model

$$y = f(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{3.1}$$

Here we assume that f follows a restricted GP of the form

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')) \tag{3.2}$$

where f is restricted to belong to the subclass of linear basis functions of the form $f(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$ with a kernel radial basis function $\boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), ..., \phi_N(\mathbf{x}))^T$ where $\phi_i(\mathbf{x}) = \kappa_r(\mathbf{x}_i, \mathbf{x})$ for some kernel κ_r (in this paper, mostly we used Gaussian radius basis kernel function, $\phi_i(\mathbf{x}) = \kappa_r(\mathbf{x}_i, \mathbf{x}) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2}}$, $i = 1, ..., N$) for the input variable $\mathbf{x}$ and a weight vector $\mathbf{w}$. To obtain a sparse model, we've employed an ARD covariance function given by $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T \mathbf{A}^{-1} \boldsymbol{\phi}(\mathbf{x}')$, where $\mathbf{A}$ is an $N \times N$ diagonal matrix with $A_{ii} = \alpha_i > 0$. Given a training

sequences $\mathbf{X}$ of size N, this GP defines a joint Gaussian:

$$p(\mathbf{f}|\mathbf{X}, \boldsymbol{\alpha}) = \mathcal{N}(\mathbf{0}, \mathbf{K}) \tag{3.3}$$

where $\mathbf{K} = \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T$ and $\boldsymbol{\Phi}$ is an $N \times N$ matrix whose i'th row is $\boldsymbol{\phi}(\mathbf{x}_i)^T$ (assuming $\boldsymbol{\Phi}$ is nonsingular by Micchellis's theorem under mild condition (Micchelli, 1984)). (Note that from the weight space viewpoint, if the prior distribution for $\mathbf{w}$ is given as $p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=1}^{N} \mathcal{N}(w_i|0, \alpha_i^{-1})$ where $\boldsymbol{\alpha}$ is a hyper-parameter vector consisted of the precisions of the noise on the weight values, which is the same with ARD prior, then the covariance function is given as $\boldsymbol{\phi}(\mathbf{x})^T\mathbb{E}(\mathbf{w}\mathbf{w}^T)\boldsymbol{\phi}(\mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T\mathbf{A}^{-1}\boldsymbol{\phi}(\mathbf{x}')$.)

We also assume that ϵ follows a variable GP given by

$$\epsilon(\mathbf{x}) \sim \mathcal{GP}(0, \beta(\mathbf{x})^{-1}\delta_{\mathbf{x},\mathbf{x}'}) \tag{3.4}$$

where $\beta(\mathbf{x})$ is a hyper-function of precision to be described in detail below and $\delta_{\mathbf{x},\mathbf{x}'}$ is a Dirac delta function with $\delta_{\mathbf{x},\mathbf{x}'} = 1$ if $\mathbf{x} = \mathbf{x}'$ and 0 otherwise. Then the GP defines the joint distribution of $\mathbf{y} = (y_1, ..., y_N)^T$ conditioned on $\mathbf{f}$ as

$$p(\mathbf{y}|\mathbf{f}, \mathbf{X}, \boldsymbol{\alpha}, \mathbf{B}) = \mathcal{N}(\mathbf{f}, \mathbf{B}^{-1}) \tag{3.5}$$

The marginal likelihood is therefore given by

$$\begin{aligned} p(\mathbf{y}|\mathbf{X}, \boldsymbol{\alpha}) &= \int p(\mathbf{y}|\mathbf{f}, \mathbf{X}, \boldsymbol{\alpha}, \mathbf{B})p(\mathbf{f}|\mathbf{X}, \boldsymbol{\alpha})d\mathbf{f} \\ &= \mathcal{N}(\mathbf{0}, \mathbf{K} + \mathbf{B}^{-1}) \end{aligned} \tag{3.6}$$

Then using the Sherman-Morrison-Woodbury matrix inversion formula, the posterior predictive distribution of $f(\mathbf{x}^*)$ for a new test point $\mathbf{x}^*$ is given by

$$p(f(\mathbf{x}^*)|\mathbf{x}^*, \mathbf{f}, \mathbf{X}, \boldsymbol{\alpha}, \mathbf{B}) = \mathcal{N}(\bar{f}(\mathbf{x}^*), \boldsymbol{\phi}(\mathbf{x}^*)^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}^*)). \tag{3.7}$$

where the mean and the covariance matrix of the posterior distribution above are given by

$$\begin{aligned} \bar{f}(\mathbf{x}^*) &= \boldsymbol{\phi}(\mathbf{x}^*)^T \mathbf{A}^{-1} \boldsymbol{\Phi}^T (\mathbf{K} + \mathbf{B}^{-1})^{-1} \mathbf{y} \\ &= \boldsymbol{\phi}(\mathbf{x}^*)^T \boldsymbol{\Sigma} \boldsymbol{\Phi}^T \mathbf{B} \mathbf{y} \end{aligned} \tag{3.8}$$

$$\boldsymbol{\Sigma} = (\mathbf{A} + \boldsymbol{\Phi}^T \mathbf{B} \boldsymbol{\Phi})^{-1} \tag{3.9}$$

and $\mathbf{B}$ is an $N \times N$ diagonal matrix with $B_{ii} = \beta(\mathbf{x}_i)$. Note that $\bar{\mathbf{f}} = \mathbf{K}(\mathbf{K} + \mathbf{B}^{-1})^{-1}\mathbf{y}$ where $\bar{\mathbf{f}} = (\bar{f}(\mathbf{x}_1), ..., \bar{f}(\mathbf{x}_N))^T$.

One distinct property is that, when the hyper-function β is given, the predictive variance, $\boldsymbol{\phi}(\mathbf{x}^*)^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}^*)$, does not depend on the target values $\mathbf{y}$ but only on input training samples $\mathbf{X}$, being unsupervised in nature. With a finite number of kernel basis functions centered on data points, the predictive variance is enlarged near densely spaced data points and small near the sparse region of data points, which is frequently observed in the equivalent kernels with local support (Hastie et al., 2009). This property is the complete opposite of the case of GPC proposed in H.-C. Kim and Lee (2007). Based on this observation, we define the variance function, $\upsilon(\mathbf{x})$ as

$$\upsilon(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}). \tag{3.10}$$

The variance function $\upsilon(\mathbf{x})$ then estimates the support region by $\{\mathbf{x} : \upsilon(\mathbf{x}) \geq \hat{\theta}\}$, where $\hat{\theta} = \min_{\mathbf{x} \in \mathbf{X}} \ \upsilon(\mathbf{x})$, where $\mathbf{X} \equiv (\mathbf{x}_1, \ldots, \mathbf{x}_N)$. (See Fig.4.3.)

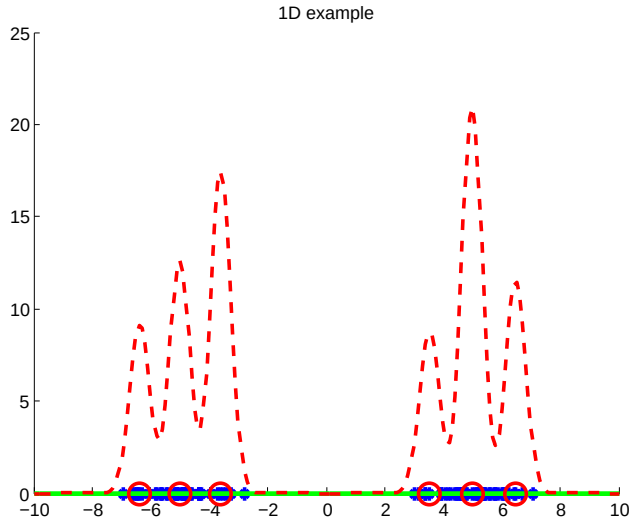


Figure 3.1 One dimensional example for proposed support function. The data points are marked by blue '*' and basis vectors are marked by red circles. Red dotted line represents the support function value. It is easily observed that the support function value is large near the basis vectors.

The performance of clustering using this support function highly depends on the choice of hyper-function $\beta(\mathbf{x})$ (or $\mathbf{B}$) because it sometimes has a positive value on only a narrow area close to a basis vector and is almost zero at a point a little distant from the basis vectors but in the cluster. In the next subsection, we suggest a method to solve this problem.

3.2.2 Determining hyper-functions and hyper-parameters

To obtain the parsimonious model for clustering using sparsity, we first determine the hyper-function $\beta(\mathbf{x})$ and then set the hyper-parameters α_i, $i = 1, \ldots, M$ used in the ARD prior, where the latter is extensively studied in the

literature and is well-known that most of α_i become infinite, thereby leading to the sparsity representation of the corresponding entries of $\mathbf{f}$ which are zero as in Tipping (2001), Wipf and Nagarajan (2008), and Williams and Rasmussen (2006). (Σ_{ij} is not zero only when neither α_i nor α_j is infinite due to the inverse of the matrix $\mathbf{A}$ and $\boldsymbol{\Sigma}$ can be contracted by discarding the rows and columns whose entries are all zero and call the corresponding input vectors for the non-infinity entries of $\boldsymbol{\alpha}$ the basis vectors.) In this subsection, we focus on how to determine $\beta(\mathbf{x})$ given other hyper-parameters α_i, $i = 1, \ldots, M$.

Following Williams and Rasmussen (2006), we define the logistic GP model for binary classification as $p(y_i|\mathbf{x}_i) = \sigma((2y_i - 1)f(\mathbf{x}_i))$ where $y_i \in \{0, 1\}$, $\sigma(z) = 1/(1+e^{-z})$ and $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ as for GP regression. If we set all training target value is zero, i.e. $\mathbf{y} = 0$, then log-posterior distribution is given by

$$\begin{aligned}
\ell(\mathbf{f}) &= \log p(\mathbf{f}|\mathbf{X}, \mathbf{y}) \\
&= \log p(\mathbf{y}|\mathbf{f}) + \log p(\mathbf{f}|\mathbf{X}) - \log p(\mathbf{y}|\mathbf{X}) \\
&= -\sum_{i=1}^{N} \{y_i \log(1 + e^{-f(\mathbf{x}_i)}) + (1 - y_i) \log(1 + e^{f(\mathbf{x}_i)})\} \\
&\quad - \frac{1}{2}\mathbf{f}^T \mathbf{K}^{-1}\mathbf{f} - \frac{1}{2} \ln |\mathbf{K}| + \text{const.}
\end{aligned}$$

Its gradient and Hessian are given by

$$\nabla \ell(\mathbf{f}) = \mathbf{y} - \boldsymbol{\sigma}(\mathbf{f}) - \mathbf{K}^{-1}\mathbf{f} \tag{3.11}$$

$$\nabla^2 \ell(\mathbf{f}) = -(\tilde{\mathbf{B}}(\mathbf{f}) + \mathbf{K}^{-1}) \tag{3.12}$$

where $\boldsymbol{\sigma}(\mathbf{f})$ is a vector whose entry $\boldsymbol{\sigma}(\mathbf{f})_i = \frac{1}{1+e^{-f(\mathbf{x}_i)}}$ and $\tilde{\mathbf{B}}(\mathbf{f})$ is a diagonal matrix whose entry $\tilde{\mathbf{B}}(\mathbf{f})_{ii} = \boldsymbol{\sigma}(\mathbf{f})_i(1 - \boldsymbol{\sigma}(\mathbf{f})_i) = \frac{e^{-f(\mathbf{x}_i)}}{(1+e^{-f(\mathbf{x}_i)^2})}$. Using iterative reweighted least squares (IRLS) to find the MAP estimate, at convergence, the

Laplace approximation of the posterior becomes

$$p(\mathbf{f}|\mathbf{X},\mathbf{y}) \approx q(\mathbf{f}|\mathbf{X},\mathbf{y}) := \mathcal{N}(\tilde{\mathbf{f}}, (\tilde{\mathbf{B}}(\tilde{\mathbf{f}}) + \mathbf{K}^{-1})^{-1}) \tag{3.13}$$

where $\tilde{\mathbf{f}}$ satisfies the equation $\tilde{\mathbf{f}} = \mathbf{K}(\mathbf{y} - \boldsymbol{\sigma}(\tilde{\mathbf{f}}))$.

The posterior predictive distribution of $f(\mathbf{x}^*)$ for a new test point $\mathbf{x}^*$ is then given by

$$p(f(\mathbf{x}^*)|\mathbf{x}^*,\mathbf{X},\boldsymbol{\alpha}) \approx \mathcal{N}(\boldsymbol{\phi}(\mathbf{x}^*)^T\tilde{\mathbf{m}}, \boldsymbol{\phi}(\mathbf{x}^*)^T\tilde{\boldsymbol{\Sigma}}\boldsymbol{\phi}(\mathbf{x}^*)) \tag{3.14}$$

where the mean vector and the covariance matrix of the posterior distribution above is given by

$$\tilde{\mathbf{m}} = \mathbf{A}^{-1}\boldsymbol{\Phi}^T\mathbf{K}^{-1}\tilde{\mathbf{f}}, \quad \tilde{\boldsymbol{\Sigma}} = (\mathbf{A} + \boldsymbol{\Phi}^T\tilde{\mathbf{B}}(\tilde{\mathbf{f}})\boldsymbol{\Phi})^{-1} \tag{3.15}$$

Thus, the clustering problem can be cast into the problem of the one-value regression or the equivalent one-class classification. The result indicates that the predictive distribution of the regression in (3.7) should be compatible with the predictive distribution of the classification in (3.14). Specifically, to have equal predictive variances of regression and classification in (3.9) and (3.15) respectively, we should have

$$\boldsymbol{\Sigma} = (\mathbf{A} + \boldsymbol{\Phi}^T\mathbf{B}\boldsymbol{\Phi})^{-1} = (\mathbf{A} + \boldsymbol{\Phi}^T\tilde{\mathbf{B}}(\tilde{\mathbf{f}})\boldsymbol{\Phi})^{-1} = \tilde{\boldsymbol{\Sigma}}$$

or equivalently

$$\beta(\mathbf{x}_i) = \mathbf{B}_{ii} = \tilde{\mathbf{B}}(\tilde{\mathbf{f}})_{ii} = \frac{e^{-\tilde{f}(\mathbf{x}_i)}}{(1 + e^{-\tilde{f}(\mathbf{x}_i)^2})} \tag{3.16}$$

Note that we set $\mathbf{y}_{class} = 0$ for the classification outputs here to make hyper function β only depend on input training data $\mathbf{X}$ since $\tilde{\mathbf{f}} = -\mathbf{K}\boldsymbol{\sigma}(\tilde{\mathbf{f}})$. In contrast, we do not need to set $\mathbf{y}_{reg} = 0$ for the regression outputs since its predictive

variance does not depend on $\mathbf{y}_{reg}$ and this flexibility of choosing any values of $\mathbf{y}_{reg}$ makes us to have equal predictive means of regression and classification in (3.8) and (3.15) respectively as follows.

$$\bar{\mathbf{f}} = \mathbf{K}(\mathbf{K} + \mathbf{B}^{-1})^{-1}\mathbf{y}_{reg} = \tilde{\mathbf{f}} = \mathbf{K}(\mathbf{y}_{class} - \boldsymbol{\sigma}(\tilde{\mathbf{f}}))$$

so that we have (by setting $\mathbf{y}_{class} = 0$)

$$\mathbf{y}_{reg} = -(\mathbf{K} + \mathbf{B}^{-1})\boldsymbol{\sigma}(\tilde{\mathbf{f}}) \tag{3.17}$$

Given this choice of hyper function β and $\mathbf{y}_{reg}$, hyper-parameter $\boldsymbol{\alpha}$ can now be found by maximizing the log-marginal likelihood function in (3.6):

$$\begin{aligned} \mathcal{L}(\boldsymbol{\alpha}) &= \ln p(\hat{\mathbf{y}}|\mathbf{X}, \boldsymbol{\alpha}) \\ &= -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\log|\mathbf{C}| - \frac{1}{2}\mathbf{y}_{reg}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{y}_{reg} \end{aligned} \tag{3.18}$$

where $\mathbf{C} = \mathbf{B}^{-1} + \mathbf{K} = \mathbf{B}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^{T}$ and $\mathbf{y}_{reg} = -\mathbf{C}\boldsymbol{\sigma}(\tilde{\mathbf{f}})$ in (3.17). Note also that all $\mathbf{K}$, $\tilde{\mathbf{f}}$, and $\tilde{\mathbf{B}}(\tilde{\mathbf{f}})$ depend on $\boldsymbol{\alpha}$. To expedite the search for optimal $\boldsymbol{\alpha}$, we adopted *sequential sparse Bayesian learning algorithm* used in Tipping and Faul (2003) which is summarized below.

Following Bishop (2006), decomposing $\mathbf{C}$ into

$$\begin{aligned} \mathbf{C} &= \mathbf{B}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^{T} = \mathbf{B}^{-1} + \sum_{m \neq i} \alpha_m^{-1}\phi_m\phi_m^{\mathrm{T}} + \alpha_i^{-1}\phi_i\phi_i^{\mathrm{T}} \\ &= \mathbf{C}_{-i} + \alpha_i^{-1}\phi_i\phi_i^{\mathrm{T}}. \end{aligned}$$

enables us to rewrite the log-marginal likelihood (3.18) as

$$\mathcal{L}(\alpha) = \mathcal{L}(\alpha_{-i}) + \lambda(\alpha_i)$$

where $\mathbf{C}_{-i}$ and $\mathcal{L}(\alpha_{-i})$ denote the matrix $\mathbf{C}$ and log-marginal likelihood with the contribution from ϕ_i removed, respectively, and

$$\lambda(\alpha_i) = \frac{1}{2}\left(\log \alpha_i - \log(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right)$$

is the term that contains all of the dependence on α_i where $s_i = \phi_i^{\mathrm{T}} \mathbf{C}_{-i}^{-1} \phi_i$ where, and $q_i = \phi_i^{\mathrm{T}} \mathbf{C}_{-i}^{-1} \phi_i$. Therefore we can maximize $\mathcal{L}(\alpha)$ over α_i by maximizing only the term $\lambda(\alpha_i)$ and α_i has an optimal solution given by

$$\alpha_i = \begin{cases} \frac{s_i^2}{q_i^2 - s_i} & \text{if} \quad q_i^2 > s_i \\ \infty & \text{if} \quad q_i^2 \leq s_i. \end{cases} \tag{3.19}$$

With this sparse solution, by the Sherman-Morrison-Woodbury matrix inversion formula the variance function becomes

$$\begin{aligned} \upsilon(\mathbf{x}) =& \boldsymbol{\phi}(\mathbf{x})^T (\mathbf{A} + \boldsymbol{\Phi}^T \mathbf{B} \boldsymbol{\Phi})^{-1} \boldsymbol{\phi}(\mathbf{x}) \\ =& \boldsymbol{\phi}(\mathbf{x})^T \mathbf{A}^{-1} \boldsymbol{\phi}(\mathbf{x}) \\ & - \boldsymbol{\phi}(\mathbf{x})^T \mathbf{A}^{-1} \boldsymbol{\Phi}^T (\mathbf{B}^{-1} + \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\Phi}^T)^{-1} \boldsymbol{\Phi} \mathbf{A}^{-1} \boldsymbol{\phi}(\mathbf{x}) \end{aligned}$$

Therefore, only the ϕ_i's corresponding to $\alpha_i \neq \infty$ remains in the variance function computations leading to sparsity. Figure 4.3 illustrates one dimensional support function generated by the proposed method. One interesting observation is that the training data points $\mathbf{x}_i$ corresponding to $\alpha_i \neq \infty$ represent the modes of the estimated support function.

3.2.3 Generalization error bound

We obtain a result with tractable complexity even in high-dimensional cases that bounds the probabilities lying outside the estimated support region of $\upsilon(\mathbf{x})$, which is similar to the result obtained in Schölkopf et al. (2001) for SVMs.

Theorem 1. *Consider a fixed but unknown probability distribution P with no atomic components on the feature space $\mathcal{F}$ with support contained in a ball of radius 1 about the origin and $\upsilon(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x})$ in (3.10). Assume that $\hat{\theta} = \min_{\mathbf{x} \in \mathbf{X}} \ \upsilon(\mathbf{x})$. Then with probability $1 - \delta$ over randomly drawn training sequences $\mathbf{X}$ of size N, for all $\gamma > 0$, and,*

$$P(\mathbf{x} : \upsilon(\mathbf{x}) < \hat{\theta} - 2\gamma) \leq \frac{2}{N} \left(K + \log \frac{N^2}{2\delta} \right),$$

where

$$K = \frac{c_1 \log(c_2 \hat{\gamma}^2 N)}{\hat{\gamma}^2} + \mathcal{D} \hat{\gamma} \log \left(e \left(\frac{(2N-1)\hat{\gamma}}{\mathcal{D}} + 1 \right) \right) + 2,$$

$c_1 = 4c^2, c_2 = ln(2)/c^2, c = 103, \hat{\gamma} = \gamma / \|\boldsymbol{\Sigma}\|$, *and* $\mathcal{D} = \mathcal{D}(\mathbf{X}, g, \hat{\theta})$

Proof. We follows the definition for $D(\mathbf{X}, f, \theta)$ as in Schölkopf et al. (2001); i.e. for a fixed $\theta \in \Re$ and a training sequence $\mathbf{X} \equiv (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, we define

$$\mathcal{D}(\mathbf{X}, f, \theta) = \sum_{\mathbf{x} \in X} d(\mathbf{x}, f, \theta).$$

where $d(\mathbf{x}, f, \theta) = \max\{0, \theta - f(\mathbf{x})\}$. First of all, there exists $\hat{\theta} = \min_{\mathbf{x} \in \mathbf{X}} \ \upsilon(\mathbf{x}) > 0$ since matrix $\boldsymbol{\Sigma}$ is positive definite. From the definition of $\upsilon(\mathbf{x})$, we note that

$$\upsilon(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}) = \mathrm{tr}(\boldsymbol{\phi}(\mathbf{x}) \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\Sigma})$$

If we let $\boldsymbol{\omega} = \mathrm{Vec}(\boldsymbol{\Sigma})$ and $\Psi(\mathbf{x}) = \mathrm{Vec}(\boldsymbol{\phi}(\mathbf{x}) \boldsymbol{\phi}(\mathbf{x})^T)$ (meaning $\Psi_{N*(i-1)+j}(\mathbf{x}) = \phi_i(\mathbf{x}) \phi_j(\mathbf{x})$) and $\omega_{N*(i-1)+j} = \Sigma_{ij}$, then it follows that $\upsilon(\mathbf{x}) = \boldsymbol{\omega} \cdot \Psi(\mathbf{x})$ and so $\upsilon(\mathbf{x})$ is a linear function in a kernel defined feature space. Then by using Theorem 17 in Schölkopf et al. (2001), with probability $1 - \delta$, and all $\gamma > 0$

$$P\{\Psi(\mathbf{x}) : \upsilon(\mathbf{x}) = \boldsymbol{\omega} \cdot \Psi(\mathbf{x}) < \hat{\theta} - 2\gamma\} \leq \frac{2}{N} \left(K + \log \frac{N^2}{2\delta} \right),$$

where

$$K = \frac{c_1 \log(c_2 \hat{\gamma}^2 N)}{\hat{\gamma}^2} + \mathcal{D} \hat{\gamma} \log \left(e \left(\frac{(2N-1)\hat{\gamma}}{\mathcal{D}} + 1 \right) \right) + 2,$$

$c_1 = 4c^2, c_2 = ln(2)/c^2, c = 103, \hat{\gamma} = \gamma / \|\boldsymbol{\Sigma}\|_F$, and $\mathcal{D} = \mathcal{D}(X, \upsilon(\mathbf{x}), \hat{\theta})$. $\square$

This result shows that the variance function of a GP characterizes the support of a high-dimensional distribution of a given data set and the estimated support set by GP has tractable complexity even in high-dimensional cases.

3.2.4 Labeling the points from the constructed support function

One interesting and distinguished feature of the proposed method is that it automatically detects the training data points that are the modes (called the representative point or exemplar) of the estimated support function or their nearby points. This feature enables us to adopt the nearest neighbor labeling algorithm naturally among other labeling methods given the estimated support function (Ben-Hur et al., 2002; J. Lee & Lee, 2005; J. Yang et al., 2002), that is to say, we assign a data point with the same label as that of the nearest representative point. Many other support-based clustering methods require to find the equilibrium points of the estimated support function as the representative points and this task is usually done by applying its associated gradient systems. The obtained representative point is normally not included in the training data set. In contrast, the proposed method automatically constructs the set of the representative points that coincide with the center of the basis vectors selected. Therefore, there is no need to find the converging mode points of the estimated support function via invoking nonlinear optimization solvers for each training data points. (See Figure 4.3.)

To assign a cluster label to the representative point, we construct an adjacency matrix A of representative points, as proposed in Ben-Hur et al. (2002);

J. Lee and Lee (2005), i.e. given a cutting level L where the set $\{\mathbf{x} : \upsilon(\mathbf{x}) \leq L\}$ characterizes the cluster structure, two representative points, say $\mathbf{x}_i, \mathbf{x}_j$, are adjacent with $A_{ij} = 1$ only if $\upsilon(\lambda\mathbf{x}_i + (1-\lambda)\mathbf{x}_j) \leq L$ for all $0 \leq \lambda \leq 1$, and $A_{ij} = 0$ otherwise. Then, we assign the same cluster label to the representative points in the same connected component generated by the adjacency matrix A. Otherwise, we can use the enhanced strategy suggested in J. Lee and Lee (2006) that characterized the cluster structure at the expense of a longer computing time. The rest of the data points are then assigned to the same cluster with the nearest representative point. The adjacency of representative points can be changed as the value of L varies; thus, the number of clusters can be controlled accordingly, as detailed in D. Lee and Lee (2010). This labeling method also has an inductive property so that any novel data point in the entire domain can be labeled accordingly.

3.3 Experimental Results

First, we applied the proposed method to the four two-dimensional toy data sets with complicated shapes mostly used in H.-C. Kim and Lee (2007) and they are shown in Figure 4.2(a) to 4.2(d). The data set in Figure 4.2(a) and 4.2(b) are from the spectral clustering website(*Spectral clustering website*, n.d.) and Figure 4.2(c) and 4.2(d) are from H.-C. Kim and Lee (2007). We used Gaussian radius basis kernel function, $\kappa_r(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}$, to cluster these data sets using the proposed method. The clustered results are presented in Figure 4.2(a') to 4.2(d'). We observed that the proposed method clustered those

data sets effectively by selecting the appropriate kernel parameters.

3.3.1 Two-dimensional toy data

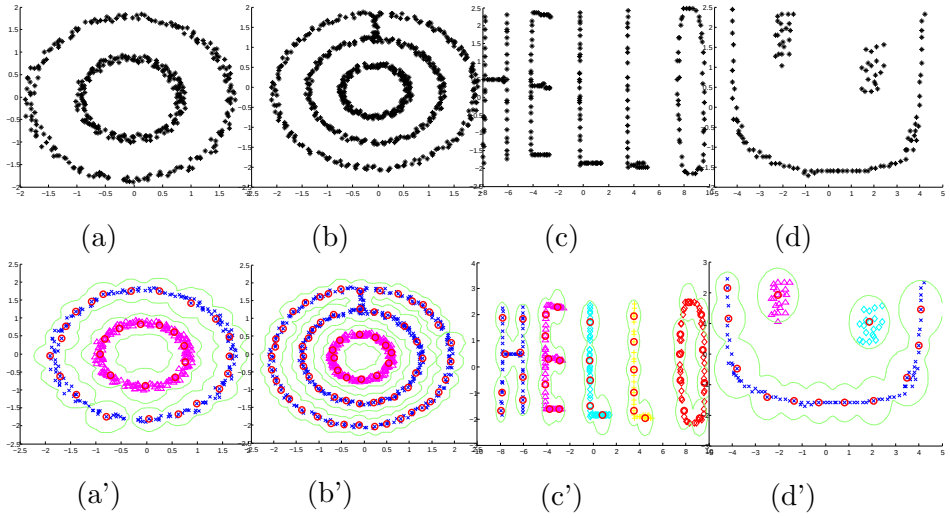


Figure 3.2 Clustering results by the proposed method. The data points are marked by black '*' and basis vectors are marked by red circles. Green lines represent the given cutting level L. (a-d) original images (a'-d') clustering results

We duplicated the data points of Figure 4.2(d) with small Gaussian noise to examine the sparsity of the proposed method. Then, we constructed the support function via the proposed method varying the number of data points. Two existing methods, namely, GPC (H.-C. Kim & Lee, 2007) and support vector clustering (SVC) (Ben-Hur et al., 2002), were also applied to construct the support functions for comparison. Figure 3.3 shows the number of basis vectors, the centered points of kernels representing the support function, for three clustering methods with different numbers of data points. We observed

that the number of basis vectors is almost invariant for the proposed methods, whereas the number of basis vectors increases as the number of data increases for SVC and GPC. The basis vectors of the proposed support function locate at the center of clusters, but the basis vectors of the other methods do not. Moreover, the graph of GPC is linear because it uses all input data points as the basis vectors. The results shown in Figure 3.3 indicate that the sparsity of the proposed method strengthens as the number of data points increases.

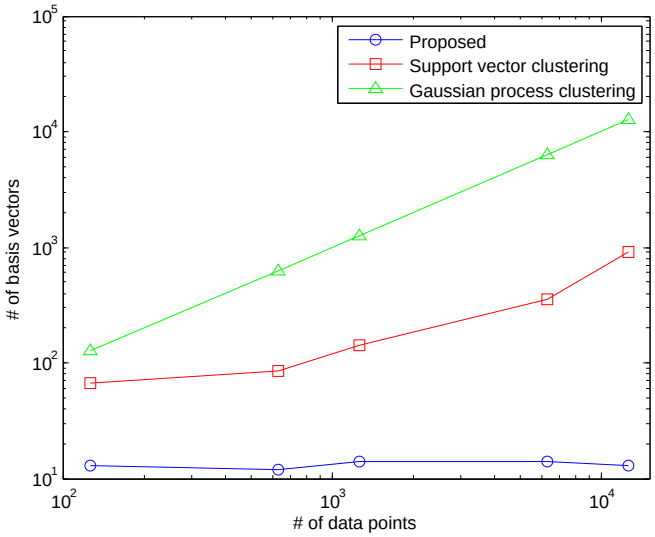


Figure 3.3 Graph of the number of basis vectors as the number of data points increases. The number of basis vectors for the proposed support function is almost invariant whereas the number of basis vectors for the other methods increase.

3.3.2 Real data sets

The proposed method was also applied to some benchmark and real data sets. The first and last data sets, Shuttle and Forest type, respectively, are from UCI repository (Asuncion & Newman, 2007). Shuttle is composed of 43,500 objects of 7 different classes with 9 variables. Forest type is compsoed of 523 instances of 4 different classes of forest types with 27 variables of both spatial and quality features. We also tested the proposed method with two handwritten digit data sets and one letter data set. USPS (Hull, 1994) data set consists of 9,298 gray-scale images of handwritten digits from 0 to 9, and the number of pixels of each image is 16 by 16. MNIST data set (LeCun et al., 1998) contains 28 by 28 gray-scale images of handwritten digits from 0 to 9. We selected 14,870 images with digits that are 0 or 1 from MNIST data set for the clustering task. OCR data set (*Spectral clustering website*, n.d.) contains 16 by 8 black and white images of handwritten lower case letters. We selected the three most frequently used letters(e, i, and n) for the task, which resulted 14,892 data points left. SVC and GPC were also adopted for the same data sets for comparison. The data sets are compared by the *adjusted Rand index* (ARI), a similarity measure between two partitions of the same data sets, and labeling time. For SVC and GPC, 1% of the data points are selected to determine the equilibrium vector via a dynamic system presented in J. Lee and Lee (2005) and the rest of data points are labeled as the same with the nearest equilibrium point. This is basically the same labeling method used in K. Kim et al. (2015).

Table 3.1 shows a comparison of clustering result of the proposed method

Table 3.1 ARI and labeling time per data. (SSC: Proposed)

	ARI			labeling time ($\times 10^{-5}$s)		
	SSC	SVC	GPC	SSC	SVC	GPC
Shuttle	0.604	0.524	0.511	3.40	9.87	769.3
USPS	0.331	0.319	0.328	1.15	51.6	5037.1
MNIST	0.988	0.812	0.987	1.48	28.7	10019
OCR	0.544	0.314	0.539	1.47	27.8	5992.1
Forest type	0.412	0.340	0.443	24.5	67.3	144.42

and existing methods. We observed that the results of the proposed method have similar or higher ARI, with significantly shorter computation time in the labeling phase.

3.3.3 Exemplar selection

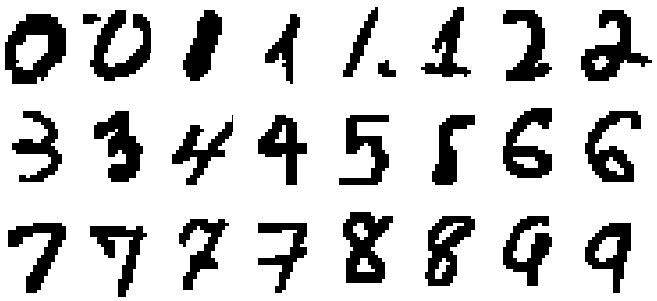


Figure 3.4 Exemplars of handwritten digits in MNIST data set.

From several results previously presented, the basis vectors are located at the center of the clusters of data points. Thus these basis vectors seem to function as exemplars. To test if this observation is correct, we applied the proposed

method to each digit of MNIST data set and determined the basis vectors. Some of these basis vectors are presented in Figure 3.4. As shown in the figure, the basis vectors were selected to represent the different shapes of each digit. Exemplars of some digits, such as 0, 3, 6, and 8, are not significantly different from each other and can be united if we change the kernel parameter. The effect of changing parameters is explained at the end of this section. However, some digits, such as 1, 2, and 7, have several extensively different shapes, and the proposed method detected those shapes.

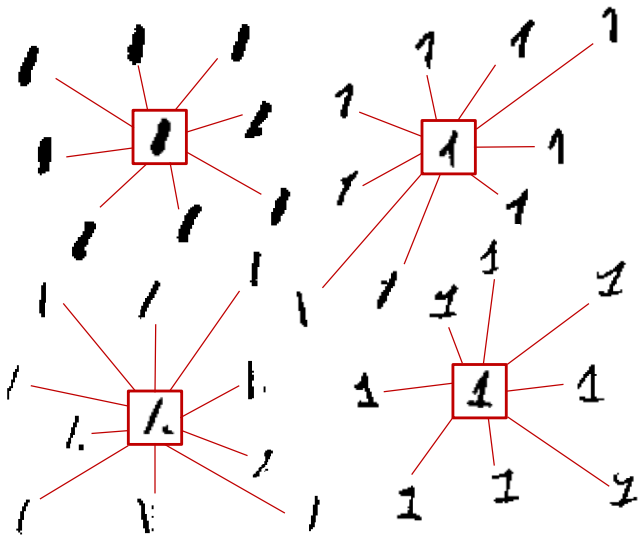


Figure 3.5 The relations between an image and its representative image. Digit 1 in MNIST data set is selected for example.

We selected digit 1 and examined which points were represented by a certain basis vector to determine whether the selected basis vector represents effectively other data points. Every data point was assigned to the nearest representative

point, and the results are shown in Figure 3.5. We observed that most of the images of the four types of digit 1 seem to be connected to their representative images. However, some of the first and third types of images do not appear to be connected to appropriate representative images because these two types take a large share among the four types of digit 1 shown in Figure 3.4, and they are extensively distributed on the feature space. This limitation of the nearest neighbor labeling algorithm considers every basis vector equally important regardless of its weight. Thus, other labeling methods using sophisticated algorithms, such as complete graph (Ben-Hur et al., 2002) or dynamic system approach (J. Lee & Lee, 2005, 2006), can overcome this problem.

3.3.4 Application to image segmentation

As a real application of clustering, the proposed method was adopted to the image segmentation task. Images for segmentation were taken from *The Berkeley segmentation dataset and benchmark* (Arbelaez et al., 2007). Each pixel of the original image was transformed into a three-dimensional vector on the color space, for example, the RGB space. The size of all images used for this task was 321 by 481. Thus the total number of pixels was 154,401. The segmentation results are shown in Figure 3.6. Figure 3.6(a) to 3.6(d) present the original images for the segmentation task and Figure 3.6(a') to 3.6(d') show the segmented images. We observed that similar colors on the images are clustered together after image segmentation.

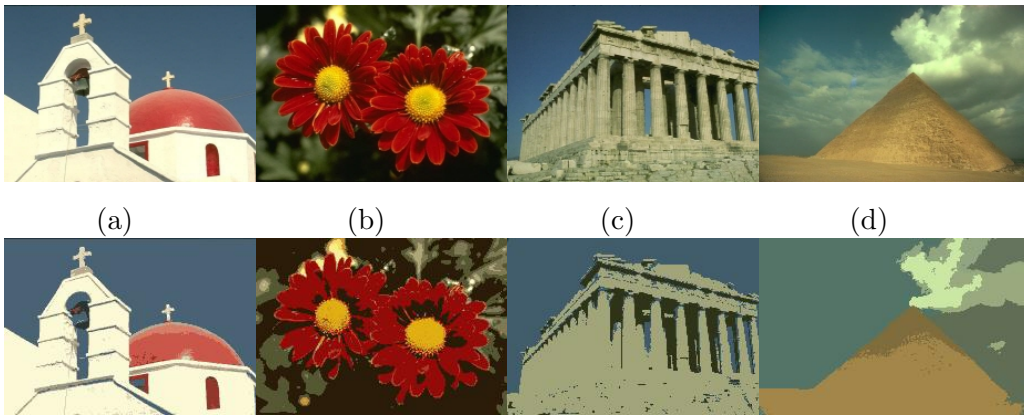


Figure 3.6 Image segmentation results (a-d) Original image (a'-d') Segmented image by the proposed method

3.3.5 Effects of the parameters

To analyze the effect of changing the Gaussian radial basis kernel function parameter, σ, on the clustering result, we adapted the proposed method to a simple data set varying the parameter where the cutting level, L, is fixed to 1.5. The results are shown in Figure 3.7. We observed that the number of basis vector increases as σ decreases. The increase in the number of basis vectors results in the separation of clusters. Particularly, as σ decreases, the cluster boundary fits the data more tightly. This characteristic coincides with that of SVC, as reported in Ben-Hur et al. (2002).

Although selecting the appropriate parameter is important to obtain the desired clustering result, the result can vary with the value of cutting level L. Figure 3.8 shows that different L values can result in different cluster assign-

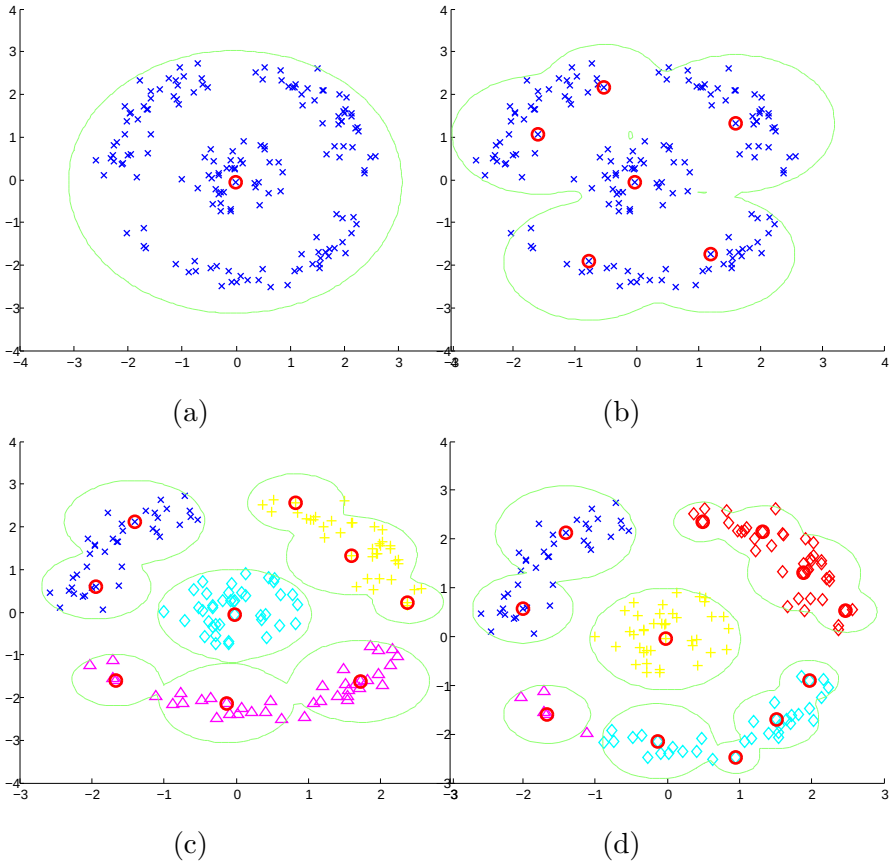
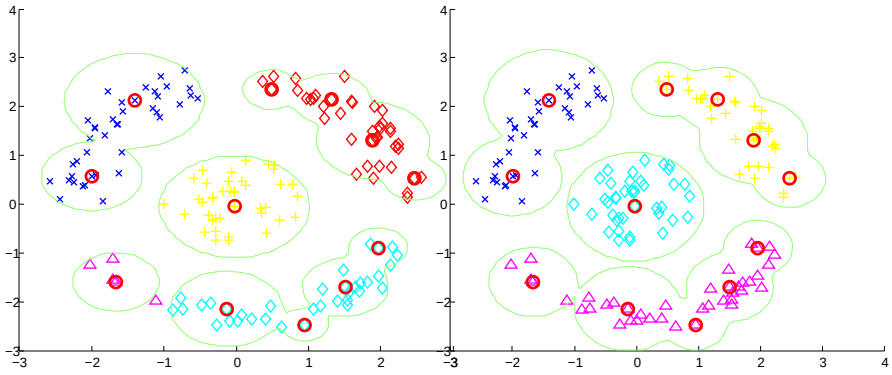


Figure 3.7 Labeling results of toy data set with varying the value of the parameter σ where the cutting level L is fixed to 1.5. (a) $\sigma = 1$ (b) $\sigma = 0.6$ (c) $\sigma = 0.42$ (d) $\sigma = 0.3$

ments. If cutting level L decreases, then the separated clusters merge if the minimum value of the support function between two clusters, that is, on the line segment between the local maximum points of one cluster and another cluster, is larger than the new cutting level. The two clusters at the bottom of Figure 3.8(a) merged in Figure 3.8(b) where the cutting level L decreased from



(a) (b)

Figure 3.8 Labeling results of toy data set with varying the value of the cutting level L where the parameter σ is fixed to 0.3. (a) $L = 1.5$ (b) $L = 0.3$

1.5 to 0.3.

3.3.6 Comparison with Other Algorithms

To find the characteristics of the proposed algorithm, we applied the proposed algorithm to the several benchmark data sets from Ultsch (2005) and compared the results with the existing clustering algorithms: k-medoids, single linkage, DBSCAN, and spectral clustering. Figure 3.9 shows the results after clustering.

From Figure 3.9, we can notice that the proposed algorthm works well for the most of the benchmark data sets. The proposed method is better to represent sophisticated distributions than the k-medoids algorithm, overlapped clusters than the single linkage algorithm, and sparse distributions than DBSCAN, the density-based algorithm, and the spectral clustering algorithm. Table 3.2 shows ARI values of these results.

Table 3.2 ARI of the proposed and other clustering algorithms for the benchmark data sets

	k-medoids	single linkage	DBSCAN	spectral	proposed
atom	0.1362	1.0000	1.0000	1.0000	1.0000
chain link	0.0867	1.0000	1.0000	1.0000	1.0000
engytime	0.8168	0.0000	0.7294	0.0000	0.4679
Lsun	0.4631	1.0000	0.9492	0.9841	1.0000
target	0.6364	1.0000	1.0000	0.7237	1.0000
tetra	1.0000	0.0000	0.6958	1.0000	1.0000
two diamonds	1.0000	0.0000	0.9457	1.0000	1.0000
wingnut	0.8269	1.0000	0.9687	1.0000	1.0000

3.4 Chapter Summary

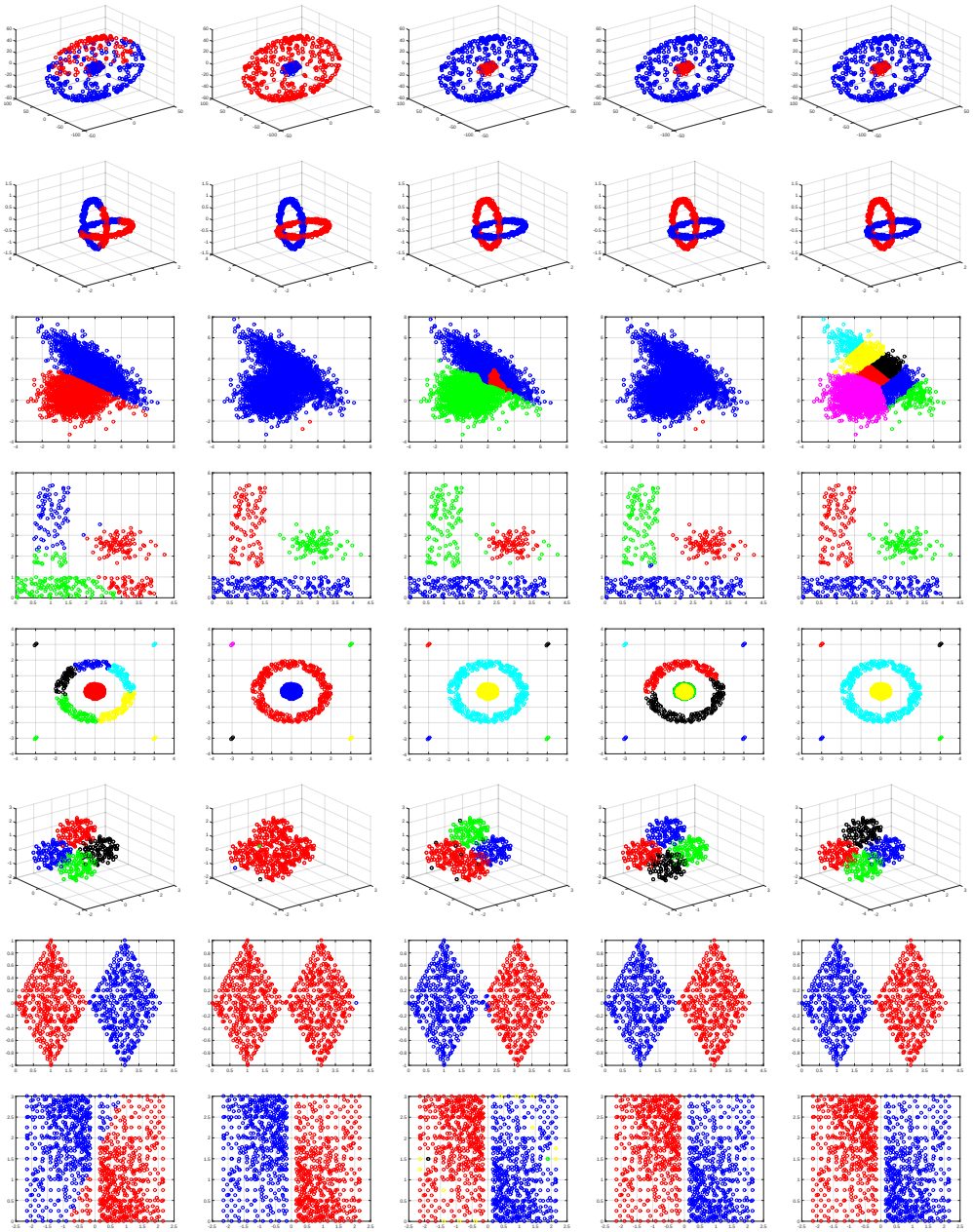
In this study, we propose a sparse support-based clustering method with a support function represented by a small number of kernel basis vectors. The method utilizes the ARD prior and the variable GP noise to build a sparse support function from the GP regression model. The method assigns the hypothetical output values (not related to the cluster labels) of the clustering data sets to obtain a tractable likelihood function in the GP regression model to determine hyper-functions and hyper-parameters efficiently. The theoretical result shows that the constructed support function can indeed estimate the support of the given data distribution.

The proposed method has several features compared with the findings of previous studies on support-based clustering methods. First, the constructed

support function is represented by a significantly smaller number of kernel center points than the other methods. Second, the kernel center points are automatically selected from the given data points during the training process and can represent the rest of the data, playing a similar role as representative points or exemplars. Third, the simple nearest neighbor method can naturally be used as the labeling method to boost the labeling in the training phase as well as the clustering in the test phase. Finally, the operability of clustering and characteristics explained previously were verified through several experiments including some benchmark and real clustering data sets, image segmentation, and handwritten digits by determining its representative data.

The proposed method still has possibilities to be improved on several points. First, in case that the precision of clustering is important, more sophisticated labeling algorithms, such as complete graph approach(Ben-Hur et al., 2002) and dynamic system approach(J. Lee & Lee, 2005, 2006; K. Kim et al., 2015), can be used for the proposed support function rather than the naïve nearest neighbor approach which concentrates on reducing labeling time. Next, although the selected parameters σ and L result in variation in the shape and number of clusters, determining appropriate parameters is sometime difficult. If a method or criterion for selecting the parameters is recommended for the proposed methods as for previous methods(K.-P. Wu & Wang, 2009), it may be helpful to use the proposed method. Finally, the number and dimension of the data set can increase in the application to some real data sets. If the algorithm of the proposed method is parallelized, then the proposed method can be effectively applied to large data sets with powerful parallel computing methods that have

been developed recently.



(a) (b) (c) (d) (e)

Figure 3.9 Clustering results with benchmark data sets of several algorithms. (a) k-medoids (b) single linkage clustering (c) DBSCAN (d) spectral clustering (e) proposed

Chapter 4

A Novel Active Learning Method for Transductive Sparse Bayesian Regression

4.1 Chapter Overview

Active learning, also called query learning or optimal experimental design, is an important branch of machine learning that builds a suitable learning model with actively provided labeled data points. Unlike semi-supervised or transductive learning, which constructs the learning model with (passively) provided labeled and unlabeled points, active learning constructs its relevant training data points with labels that are reductively essential in efficient learning and asks the user or the membership counselor to label them; for example, active learning makes queries for data to be labelled. Hence, active learning can play a very important role when the cost of obtaining the labels of the data points is expensive, as in the case of using commercial text, speech, or video class labels, which should be annotated by a human expert(Lang, 1995; Zhu et al., 2005; Settles & Craven, 2008; Zha et al., 2012), scientific or engineering results from complicated experiments(Flaherty et al., 2005; J. Guo et al., 2004; King

et al., 2004, 2009; Liu, 2004), or market research or survey results often used in business and the social sciences.

In the last decade, many active learning strategies have been developed for the state-of arts kernel machines such as SVMs and GPs. They are mostly based on querying the most uncertain points among the unlabeled data pool and include multikernel SVR (Ceperic et al., 2012), ε-SVR (Ceperic et al., 2014), active SVR (Demir & Bruzzone, 2014), kernel ridge regression (Douak et al., 2013), Bayesian ridge kernel regression (Paisley et al., 2010), and GP regression (Seo et al., 2000) and (Krause & Guestrin, 2007). Comparatively, there have been very few studies on active learning strategies for sparse Bayesian learning models such as RVM, despite of its recent explosion of interest(Liu, 2004; Naveen, 2012; Matsumoto & Hori, 2014; Ribeiro et al., 2006; Sabuncu et al., 2014; Shuib et al., 2014; Tipping, 2001). Silva and Ribeiro (2007) proposed an active learning procedure for RVM by querying the label of the furthest point from the relevance vectors with its application to text classification. Paisley et al. (2010) suggested another active learning procedure to find the optimal queries. However, this procedure requires a different procedure for active learning to make queries for labeling and build the final model using RVM.

In this chapter, we propose a novel active learning algorithm for sparse Bayesian regression. To this end, we first develop a transductive and generalized version of RVM regression wherein the basis of the model can be selected from the unlabeled data points as well as the labeled data points. Then, we propose an active learning strategy that can make queries for labeling using only the relevance vectors automatically determined from the developed model, thereby

making it unnecessary to require an additional procedure for active learning.

The remainder of this chapter is organized as follows. In section 2, we propose a transductive GRVM with an active learning procedure and present its algorithm and implementation. Section 3 gives the experimental results of both artificial data sets and real data sets. Finally, we make concluding remarks and cite the future directions of this research in section 4.

4.2 Proposed method

RVM, which is proposed by Tipping (2001), is a sparse Bayesian learning method based on ARD prior on its weights and is successively applied to various regression and classification tasks. One of the advantages of using RVM over other kernel methods, such as SVM and GP regression, is its sparseness with comparable performances and computational costs (Naveen, 2012; Shuib et al., 2014).

Here, we propose a transductive and generalized version of the RVM regression, namely a transductive GRVM, which can be used when only a small portion of the data points are labeled. Then, we suggest three querying strategies that exploit the characteristics of transductive GRVM. After obtaining the labels of queried data points, the proposed algorithm repeats training the model with the new labeled data sets and querying other data points until the stopping criterion is satisfied.

4.2.1 Transductive sparse Bayeasin regression

The aim of the transductive GRVM suggested in this section is to construct GRVM model which uses both labeled and unlabeled data points and selects the relevance vectors from both of them.

Like other regression models using basis functions, we consider a Bayesian regression of the form:

$$y = f(\mathbf{x}) + \epsilon \tag{4.1}$$

where ϵ is a noise with mean zero and variance σ^2 that are uncorrelated with data (we are assuming a more general case than a Gaussian noise $\mathcal{N}(0, \sigma^2)$) and $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})$ where $\mathbf{w}$ is a weight vector and $\boldsymbol{\phi}(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_N(\mathbf{x})]^{\mathrm{T}}$ is a vector of basis functions. In this paper, we use a kernel function $\kappa(\mathbf{x}, \mathbf{x}')$ as a basis function where the i'th basis function is given by $\phi_i(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}_i)$. $\phi_0(\mathbf{x}) = 1$ is sometimes included in the basis vector to represent the bias term.

Now let $\{\mathbf{X}_L, \mathbf{y}_L\}$ be the labeled data set and $\{\mathbf{X}_U\}$ be the unlabeled data set. We also assume that there are L labeled data points and U unlabeled data points, and the total number of data points is $N = L+U$. Rhe regression model can be represented as

$$\mathbf{y}_L = \boldsymbol{\Phi}_{L,L+U}\mathbf{w} + \boldsymbol{\epsilon}_L \tag{4.2}$$

where $\boldsymbol{\Phi}_{L,L+U}$ is an $L \times (L+U)$ matrix composed of kernel values of labeled points and the whole points whose ij'th element is $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ where $\mathbf{x}_i$ is selected from the labeled data points, i.e. the rows of $\mathbf{X}_L$, and $\mathbf{x}_j$ is selected from the whole points, i.e. the rows of $\mathbf{X} = [\mathbf{X}_L^{\mathrm{T}}\ \mathbf{X}_U^{\mathrm{T}}]^{\mathrm{T}}$. Here, $\mathbf{w}$ is an $N \times 1$ weight vector and $\boldsymbol{\epsilon}_L$ is an $L \times 1$ noise vector with zero mean vector and covariance matrix

$\sigma^2\mathbf{I}_L$ where $\mathbf{I}_L$ refers to the L dimensional identity matrix.

To obtain a sparse solution, we employ a generalized version of an ARD prior as in Tipping (2001) for the weight vector, i.e. $p(\mathbf{w}|\mathbf{A})$ has zero mean vector and $N \times N$-covariance matrix $\mathbf{A}^{-1}$) with the diagonal matrix $\mathbf{A} = \mathrm{diag}\{A_{ii}\}$ as the hyperparameter. These hyperparameters, the ARD prior $\mathbf{A}$ and the noise σ^2, can be found by type-II maximum likelihood (empirical Bayes) estimation.

Since it is difficult in general to obtain the exact marginal likelihood function conditional on $\mathbf{X}$, $\mathbf{A}$, and σ^2, we derive it by applying the Laplace approximation to the joint distribution of labeled outputs and unlabeled predictive outputs. Notice that both labeled outputs, $\mathbf{y}_L$ and unlabeled predictive outputs, $\mathbf{f}_U = \mathbf{\Phi}_{U,L+U}\mathbf{w}$, a vector of $f(\mathbf{x})$ in (5.16) for the unlabeled data points where $\mathbf{\Phi}_{U,L+U}$ is a $U \times (L+U)$ kernel value matrix of unlabeled points and the whole points, have zero mean vectors and the covariance matrix of $\mathbf{y}_L$ and $\mathbf{f}_U$ conditional on $\mathbf{X}$, $\mathbf{A}$, and σ^2 is given by

$$\begin{pmatrix} \mathrm{cov}(\mathbf{y}_L, \mathbf{y}_L) & \mathrm{cov}(\mathbf{y}_L, \mathbf{f}_U) \\ \mathrm{cov}(\mathbf{f}_U, \mathbf{y}_L) & \mathrm{cov}(\mathbf{f}_U, \mathbf{f}_U) \end{pmatrix}$$

The covariances conditional on $\mathbf{X}$, $\mathbf{A}$, and σ^2 can now be obtained by applying the formula of conditional expectations as follows.

$$\begin{aligned} \mathrm{cov}(\mathbf{y}_L, \mathbf{y}_L) &= \mathbb{E}[\mathbf{y}_L\mathbf{y}_L^{\mathrm{T}}] = \mathbb{E}_{\mathbf{w}}\left[\mathbb{E}[(\mathbf{\Phi}_{L,L+U}\mathbf{w} + \boldsymbol{\epsilon}_L)(\mathbf{\Phi}_{L,L+U}\mathbf{w} + \boldsymbol{\epsilon}_L)^{\mathrm{T}}|\mathbf{w}]\right] \\ &= \mathbf{\Phi}_{L,L+U}\mathbb{E}_{\mathbf{w}}[\mathbf{w}\mathbf{w}^{\mathrm{T}}]\mathbf{\Phi}_{L,L+U}^{\mathrm{T}} + \sigma^2\mathbf{I}_L \\ &= \mathbf{\Phi}_{L,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{L,L+U}^{\mathrm{T}} + \sigma^2\mathbf{I}_L \end{aligned}$$

$$\begin{aligned} \mathrm{cov}(\mathbf{y}_L, \mathbf{f}_U) &= \mathbb{E}[\mathbf{y}_L\mathbf{f}_U^{\mathrm{T}}] = \mathbb{E}_{\mathbf{w}}\left[\mathbb{E}[(\mathbf{\Phi}_{L,L+U}\mathbf{w} + \boldsymbol{\epsilon}_L)(\mathbf{\Phi}_{U,L+U}\mathbf{w})^{\mathrm{T}}|\mathbf{w}]\right] \\ &= \mathbf{\Phi}_{L,L+U}\mathbb{E}_{\mathbf{w}}[\mathbf{w}\mathbf{w}^{\mathrm{T}}]\mathbf{\Phi}_{U,L+U}^{\mathrm{T}} = \mathbf{\Phi}_{L,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{U,L+U}^{\mathrm{T}} \end{aligned}$$

$$\begin{aligned}\mathrm{cov}(\mathbf{f}_U, \mathbf{f}_U) &= \mathbb{E}[\mathbf{f}_U \mathbf{f}_U^{\mathrm{T}}] = \mathbb{E}_{\mathbf{w}}\left[\mathbb{E}[(\mathbf{\Phi}_{U,L+U}\mathbf{w})(\mathbf{\Phi}_{U,L+U}\mathbf{w})^{\mathrm{T}}|\mathbf{w}]\right] \\ &= \mathbf{\Phi}_{U,L+U}\mathbb{E}_{\mathbf{w}}[\mathbf{w}\mathbf{w}^{\mathrm{T}}]\mathbf{\Phi}_{U,L+U}^{\mathrm{T}} = \mathbf{\Phi}_{U,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{U,L+U}^{\mathrm{T}}\end{aligned}$$

After applying the Laplace approximation, we obtain the following approximated joint distribution of $\mathbf{y}_L$ and $\mathbf{f}_U$ conditional on $\mathbf{X}$, $\mathbf{A}$, and σ^2.

$$\begin{pmatrix}\mathbf{y}_L \\ \mathbf{f}_U\end{pmatrix} \approx \mathcal{N}\left(\mathbf{0}, \begin{pmatrix}\mathbf{\Phi}_{L,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{L,L+U}^{\mathrm{T}} + \sigma^2\mathbf{I}_L & \mathbf{\Phi}_{L,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{U,L+U}^{\mathrm{T}} \\ \mathbf{\Phi}_{U,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{L,L+U}^{\mathrm{T}} & \mathbf{\Phi}_{U,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{U,L+U}^{\mathrm{T}}\end{pmatrix}\right) \tag{4.3}$$

Therefore, its corresponding approximated marginal likelihood function of $\mathbf{y}_L$ is given by

$$\begin{aligned}p(\mathbf{y}_L|\mathbf{X}_L, \mathbf{X}_U, \mathbf{A}, \sigma^2) &= \int p(\mathbf{y}_L, \mathbf{f}_U|\mathbf{X}_L, \mathbf{X}_U, \mathbf{A}, \sigma^2)\mathrm{d}\mathbf{f}_U \\ &\approx \mathcal{N}(\mathbf{0}, \mathbf{\Phi}_{L,L+U}\mathbf{A}^{-1}\mathbf{\Phi}_{L,L+U}^{\mathrm{T}} + \sigma^2\mathbf{I}_L). \end{aligned} \tag{4.4}$$

In maximizing the likelihood in (4.4) with respect to the hyperparameters $\mathbf{A}$ and β, most of the entries of $\mathbf{A}$ become infinite as in the traditional RVM; thus only a small portion of data points are selected as relevance vectors. However, those relevance vectors are also chosen from the unlabeled data points as well as the labeled ones because the basis vector includes the kernel functions centered at the unlabeled data points.

Because of the ARD prior, it can be shown as described in Tipping (2001) and Tipping and Faul (2003) that most of diagonal elements in $\mathbf{A}$ become infinite as a result of likelihood maximization and the corresponding weights become zero. Thus, the regression model in (5.16) can be represented only by a small number of kernel functions relevant to the input points corresponding to the nonzero weights. These points are called *r*elevance vectors.

Finally, our transductive version of sparse Bayesian regression is given by the posterior predictive distribution for a new point $\mathbf{x}^*$ which can be derived using the Sherman-Morrison-Woodbury formula as follows:

$$p(y^*|\mathbf{x}^*, \mathbf{y}_L, \mathbf{X}_L, \mathbf{X}_U, \mathbf{A}, \beta) \sim \mathcal{N}(m_f(\mathbf{x}^*), \sigma^2 + \sigma^2 \boldsymbol{\phi}(\mathbf{x}^*)^{\mathrm{T}} \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}^*)) \tag{4.5}$$

where

$$m_f(\mathbf{x}^*) = \boldsymbol{\phi}(\mathbf{x}^*)^{\mathrm{T}} \boldsymbol{\Sigma}_{L+U} \boldsymbol{\Phi}_{L,L+U}^{\mathrm{T}} \mathbf{y}_L \tag{4.6}$$

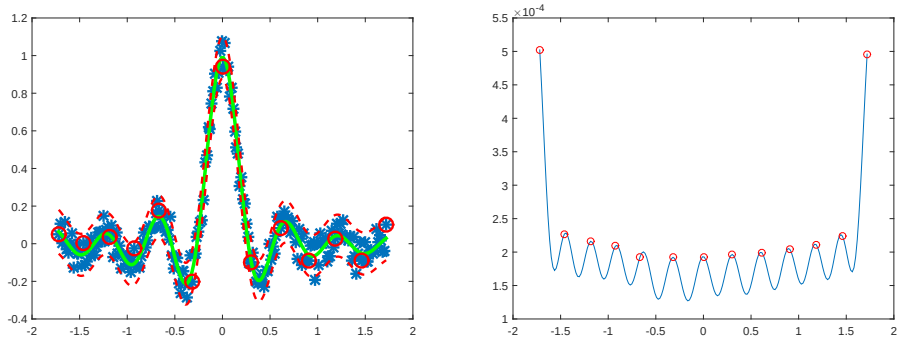
$$\boldsymbol{\Sigma}_{L+U} = (\sigma^2 \mathbf{A} + \boldsymbol{\Phi}_{L,L+U}^{\mathrm{T}} \boldsymbol{\Phi}_{L,L+U})^{-1}. \tag{4.7}$$

One notable fact here is that the final model does not involve the computations of the kernel values between the unlabeled points.

4.2.2 Active Learning Strategy

Our key idea for active learning strategy is based on the observations that the obtained relevance vectors are located at the local maximal points of the predictive variance or near them. Figure 4.1 illustrates this behavior by showing 1D example generated by the proposed transductive regression. The predictive mean and 95% confidence interval estimated by the predictive variance are denoted by a green line and red dotted lines, respectively, in Figure 4.1-(a). The relevance vectors are denoted by red circles. Figure 4.1-(b) represents the predicted variance, subtracting the common value σ^2, with a blue line and the relevance vectors with red circles. In this figure we can observe that the most of the relevance vectors are selected at the extreme points, local optimal points and boundary points, of the predictive mean function and the variance

values have the local maximal values at these relevance vectors. This observation can be inferred from the fact that the covariance matrix $\mathbf{\Sigma}$ in (5.20) has zero values except the entries with both row and column indices corresponding to the nonzero weights, or non-infinite values of A_{ii}. Then the first term in variance, $\boldsymbol{\phi}(\mathbf{x}^*)^{\mathrm{T}}\mathbf{\Sigma}_{\mathbf{L+U}}\boldsymbol{\phi}(\mathbf{x}^*)$, has a larger value near the relevance vector because $\boldsymbol{\phi}(\mathbf{x}^*)$ has a larger value for the entries corresponding to nonzero values of $\mathbf{\Sigma}_{L+U}$.



(a) (b)

Figure 4.1 1D example of RVM regression. (a) Predictive mean is denoted by a green line and 95% confidence interval is denoted by red dotted lines. The relevance vectors are denoted by red circles. (b) Variances without the common term (σ^2). The relevance vectors are denoted by red circles.

Based on this property, we propose the following querying strategies for active learning with no additional calculation.

- **Querying all unlabeled relevance vectors:** Our first strategy is simply querying all of the unlabeled relevance vectors. This strategy does not require any other calculation and makes queries for the labels of the obtained relevance vectors.

- **Querying the most uncertain relevance vector:** The second strategy is querying the most uncertain point among the unlabeled relevance vectors. It coincides with the regression version of the traditional uncertainty sampling which considers the point with the largest variance as the most uncertain point. This strategy involves a simple additional computation for the variance of the unlabeled relevance vectors, not the whole data points.

- **Querying the farthest relevance vector:** Our last querying strategy is similar to the second one. However, this strategy selects one unlabeled relevance vector with the highest minimal distance to the labeled points is maximum among the unlabeled relevance vectors, i.e.

$$\mathbf{x}_{\text{query}} = \underset{\mathbf{x} \in \mathcal{RV}_U}{\operatorname{argmax}} \min_{\mathbf{x}_l} \operatorname{dist}(\mathbf{x}, \mathbf{x}_l) \tag{4.8}$$

  where $\mathcal{RV}_U$ is the set of unlabeled relevance vectors, $\mathbf{x}_l$ is selected from the labeled data set, and $\operatorname{dist}(\mathbf{x}_1, \mathbf{x}_2)$ is a pre-defined distance function between two points $\mathbf{x}_1$ and $\mathbf{x}_2$. Although this strategy requires an additional computation for calculating the distances, it prevents querying some redundant points located close to the labeled points and selects only the points far from the given labeled points to give new information to the model.

Among these three strategies, the first one involves the smallest number of iterations because it requires the labels of several data points at once. However, the limitation of this strategy is that it may require large cost in the case when the labeling cost is expensive.

4.2.3 Algorithm and Implementations

The proposed method first trains the proposed transductive GRVM model with a given initially labeled and unlabeled data sets. After the training phase, the method requires the labels of querying points selected by the querying strategy given in section 4.2.2. The detailed procedure for the proposed method is now given as follows.

There are some implementation issues that need to be addressed. First, to implement the proposed transductive GRVM, we used *Probabilistic modeling toolkit*(Murphy & Dunham, 2008) and the fast marginal likelihood maximization proposed in Tipping and Faul (2003) to reduce computational complexity. The naïve implementation requires $O(N^3)$ computations to find hyperparameters where N is the number of training instances. Meanwhile, the fast marginal likelihood maximization algorithm (Tipping & Faul, 2003) requires $O(M^3)$ where M is the number of relevance vectors included in the model by adding and removing the candidate basis vector by one at once. M can be N at its maximum, however, these values appear to be very smaller in practice. Therefore, the complexity of the proposed method is of $O(cM^3)$ complexity where c and M refer to the maximum number of iterations the maximum number of relevance vectors included in the model during the iteration, respectively.

When the user chooses the initial selection of the labeled data points, one can select L_1 points randomly from the whole data set $\mathcal{D}$ which is the simplest approach requiring no extra calculation. However, the performance and convergence speed can be very sensitive with the choice of initial selections.

Algorithm 1 Active learning for transductive GRVM

(A1. Initialization:)

1: Select data points which are labeled initially from given data points $\mathcal{D} = \{\mathbf{x}_k\}_{k=1}^{N}$.

2: Construct the transductive RVM model proposed in section 4.2.1 with the initially labeled data points $\mathcal{D}_L^1 = \{\mathbf{x}_k, y_k\}_{k=1}^{L_1}$ and remaining unlabeled data points $\mathcal{D}_U^1 = \{\mathbf{x}_k\}_{k=L_1+1}^{N}$.

(A2. Construct the active learning model:)

1: Set $j = 1$.

2: **repeat**

3: Find the set of unlabeled relevance vectors $\mathcal{RV}_U$ and select the querying set $\mathcal{D}_q$ based on the querying strategy presented in section 4.2.2..

4: **if** querying all unlabeled relevance vector **then**

5: $\mathcal{D}_q := \mathcal{RV}_U$.

6: **else if** querying the most uncertain relevance vector **then**

7: $\mathcal{D}_q := \{\mathbf{x} : \mathbf{x} = \text{argmax}_{\mathbf{x}_u \in \mathcal{RV}_U} \text{Var}(\mathbf{x}_u)\}$.

8: **else if** querying the farthest relevance vector **then**

9: $\mathcal{D}_q := \{\mathbf{x} : \mathbf{x} = \text{argmax}_{\mathbf{x} \in \mathcal{RV}_U} \min_{\mathbf{x}_l \in \mathcal{D}_L^j} \text{dist}(\mathbf{x}, \mathbf{x}_l)\}$.

10: **end if**

11: $\mathcal{D}_L^{j+1} := \mathcal{D}_L^j \bigcup \{\mathbf{x}_q, y_q\}_{\mathbf{x}_q \in \mathcal{D}_q}$ and $\mathcal{D}_U^{j+1} := \mathcal{D}_U^j \setminus \mathcal{D}_q$.

12: Construct the transductive RVM model with new labeled data points, $\mathcal{D}_L^{j+1}$, and unlabeled points, $\mathcal{D}_U^{j+1}$.

13: $j \leftarrow j + 1$.

14: **until** the stopping criterion is satisfied.

15: Save the relevance vectors and the corresponding weights of the final model.

Clustering-based approach can be a good alternative. This approach first clusters the whole data points and selects a few points from each cluster. This approach may reflect the distribution of data points thereby being used for pursuing higher performance and faster convergence, although it requires one

additional clustering phase. Any other sampling method can also be applied in addition.

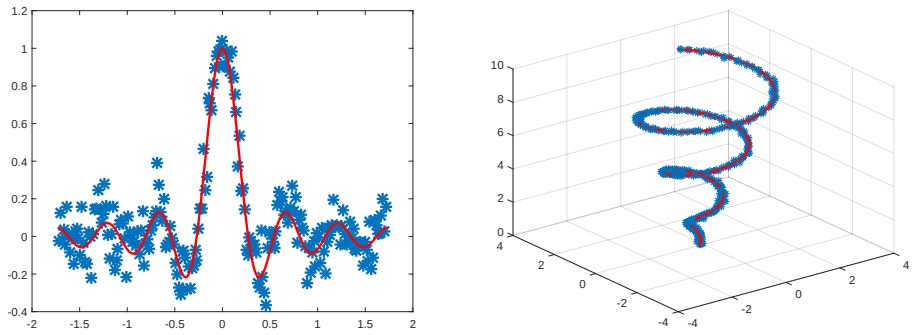
If all relevance vectors are selected from the labeled points, the model converges and the algorithm stops. At times, some stopping criteria other than convergence might be required for practical reasons, including labeling cost and computational time. In this case, we can give a few suggestions. The first criterion is to set the maximum number of iterations and stop the algorithm if the number of iteration exceeds that number. The second is to stop the algorithm when the ratio of labeled points exceeds the pre-set value. The last one is to stop the algorithm if the number of relevance vectors becomes higher than some pre-set tolerance. The combination of these methods can also be used for the stopping criterion.

4.3 Experimental Results

4.3.1 Toy data sets

To verify the performance of the algorithm, we first applied the proposed algorithm to two artificial data sets, *sinc* data set with a curved shape and *spiral* data set with a linear shape. Figure 4.2 illustrates these two data sets where blue stars and red lines denote the noise-additive points used for the experiments and the noise-free true values, respectively.

In our experiment, Gaussian kernel, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/2\sigma^2\}$, is used for the basis kernel function where the kernel parameter σ is determined by a 10-fold cross-validation result using the whole data points. For the

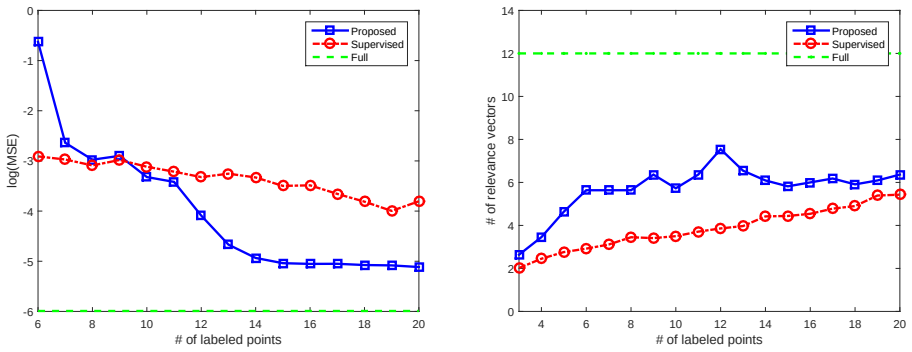


(a) (b)

Figure 4.2 Toy data sets for the proposed method. Red lines denote the true values and blue stars are noise-additive data points. (a) *sinc* data set (b) *spiral* data set

initialization of the proposed method, we randomly selected three initial labeled points. Next, we used the *querying the farthest relevance vector* strategy proposed in section 4.2.2 and stopped the algorithm if the number of labeled points exceeded 20, i.e., 10% of the whole data points. The results are shown in Figure 4.3. Figure 4.3(a) indicates the logarithm value of mean squared error (MSE) and Figure 4.3(b) shows the number of relevance vectors. Proposed, symbolized by the blue solid lines with squares, refers to the average results of the proposed method with 10 times repetition, whereas Supervised, represented by red dashed-dot lines with circles, refers to the results of RVM using the same number of randomly selected labeled points as that of the proposed method. For each number of labeled points, we averaged the results of 100 random selections for Supervised result. Standard errors of both Proposed and Supervised results are not presented because these are too small to be noticed compared

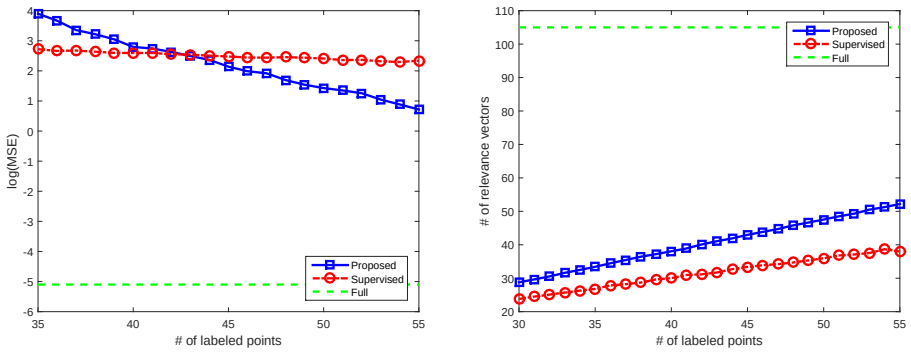
to their means. In addition, Full, denoted by green dashed lines, is the result of a general RVM applied to the whole data points that are all labeled.



(a) (b)

Figure 4.3 Simulation results of *sinc* data set. (a) logarithm values of MSE (b) number of relevance vectors.

In Figure 4.3, the proposed method overwhelms the random selection after a few number of iterations for *sinc* data set with the comparable number of relevance vectors. The MSE values of the proposed method and random selection at $L = 20$ are $(6.01 \pm 0.69) \times 10^3$ and $(2.23 \pm 2.31) \times 10^{-2}$, respectively. The random selection model has a large standard error because random selection sometimes fails to construct the robust model. Figure 4.3 is shown from the fourth iteration, i.e. L=6, for visibility since the results of proposed methods are not robust for a few first iterations, hence their MSE values are very high. The MSE of the proposed method is expectedly larger than that of the full RVM since the number of relevance vectors used for labeling in the proposed method is smaller than that used for the full method.

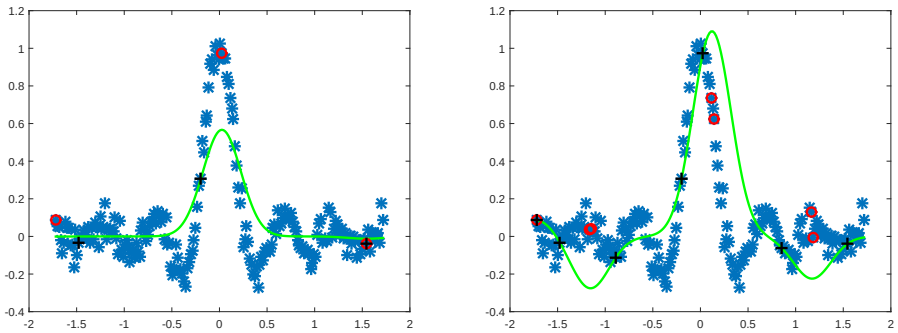


(a) (b)

Figure 4.4 Simulation results of *spiral* data set. (a) logarithm values of MSE (b) number of relevance vectors.

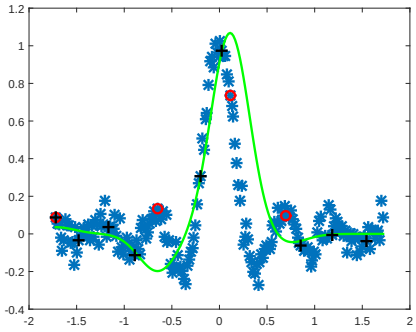
Figure 4.4 presents the simulation results of *spiral* data set. Figure 4.4(a) shows logarithm values of MSE and Figure 4.4(b) indicates the number of relevance vectors. The initial number of labeled points is 30 and the iterations proceed until the number of labeled points does not exceed 55, which is almost 10% of the given data points. *Querying the farthest relevance vector* strategy is also used for the querying strategy. The proposed method results in significantly lower MSE value than the random selection, 2.05 ± 0.80 and 10.36 ± 2.00, whereas both their MSEs are higher than that of the full RVM, 0.0061.

Figure 4.5 illustrates changes of the model through iterations. Blue stars denote the original input points, red circles signify the selected relevance vectors, and black plus signs represent the labeled points. The predictive model constructed in each iteration is denoted by the green line. As the iteration step proceeds, the model changes to fit the data points and the number of labeled

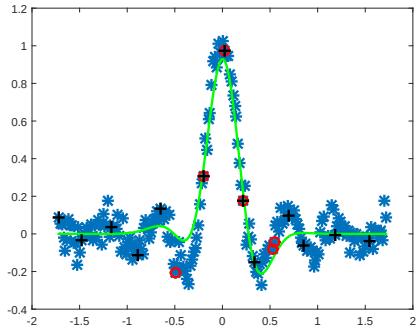


(a)

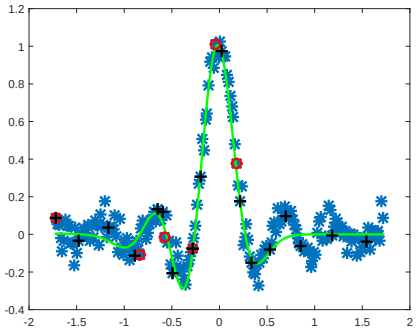
(b)



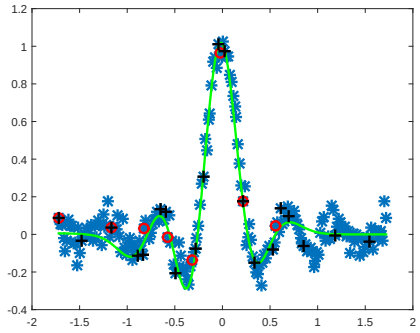
(c)



(d)



(e)



(f)

Figure 4.5 Example of changing models for *sinc* data set with (a) $L = 3$ (b) $L = 7$ (c) $L = 9$ (d) $L = 13$ (e) $L = 17$ (f) $L = 20$. Given data points, labeled points, and relevance vectors are denoted by blue '*', black '+', and red 'o' respectively. The predictive mean of the model is represented by the green line.

points increases with the selected querying strategy. As discussed earlier, we can observe that most of the relevance vectors are located near the extreme position, hence, the selected labeled vectors are also located at these positions.

4.3.2 Real data sets

To verify the performance of the proposed method for real applications, we next applied the proposed method to four real data sets from the UCI repository(Asuncion & Newman, 2007) (*abalone*, *airfoil*, *concrete*, and *wine-red*) and another data set of American-type put option prices of S&P 100 index (*Am. option*). The basic characteristics of these data sets are shown in Table 4.1.

Table 4.1 Basic characteristics of data sets

Data set	# of instances	# of dimensions	Output range
abalone	4177	8	1-29
airfoil	1503	5	103.38-140.99
concrete	1030	8	2.33-82.60
wine-red	1599	11	3-8
Am. option	6954	2	1.03-125.95

The comparison results for each of the three suggested querying strategies are shown in Table 4.2 to 4.4 respectively. In the Proposed column, we reported the averages and standard errors of MSE, number of relevance vectors, and number of labeled points for at least 20 converged results for the proposed method. The convergence of model means that the all relevance vectors are selected from the labeled point set so there are no remaining points to be

queried. We also limited the maximum number of labeled points to 50% of the whole data points. In the Supervised column, we reported the averaged results of random selections where the predictive models are constructed with the same number of randomly-selected labeled points as that of the proposed algorithm. For each number of labeled points, the random selection was repeated at least 100 times and its results were averaged so that they can be compared with the proposed method. Here, superscripts * and ** are given to the data if the result of Proposed applied to it was significantly better than that of Supervised by Wilcoxon signed-rank test with the level of significance α set to 0.05 and 0.01 respectively.

As is shown in the Tables 4.2 to 4.4, the proposed algorithm performed significantly better than the random selection regardless of the querying strategy for all the used data sets. In addition, all these results are statistically significant, although we observed some different characteristics among the applied three querying strategies. The second querying strategy in Table 4.3, *querying the most uncertain relevance vector*, has converged with smaller number of labeled points than the other strategies; however, it has higher standard error than the other strategies, implying that the constructed models converge differently depending on its initial selection of labeled points.

The first querying strategy in Table 4.2, *querying all unlabeled relevance vectors*, and the last strategy in Table 4.4, *querying the farthest relevance vector*, usually result in smaller variance than the other, whereas they converge with a large number of labeled data. Based on these results, it appears that the models constructed by the proposed method, with a variety of initial selections, become

Table 4.2 Experimental results for data sets. *Querying all unlabeled relevance vectors* strategy is used for the querying strategy. # of RV refers to the number of relevance vectors and L is the averaged number of labeled points that the proposed algorithm converges.

Data set	Proposed			Supervised		Full	
	MSE	# of RV	L	MSE	# of RV	MSE	# of RV
abalone	$6.72 \pm 0.76^{**}$	6.45	38.35	7.85 ± 0.62	4.44	4.23	26
airfoil	$7.28 \pm 0.41^{**}$	105.35	709.95	14.64 ± 0.98	98.05	4.96	122
concrete	$70.06 \pm 14.08^{**}$	25.85	159.75	80.09 ± 14.38	21.56	22.50	63
wine-red	$0.78 \pm 0.21^{**}$	8.85	41.95	0.94 ± 0.13	5.67	0.36	33
Am. option	$3.81 \pm 2.65^{**}$	12.20	86.45	24.63 ± 6.73	10.06	1.27	25

Table 4.3 Experimental results for data sets. *Querying the most uncertain relevance vector* strategy is used for the querying strategy. # of RV refers to the number of relevance vectors and L is the averaged number of labeled points that the proposed algorithm converges.

Data set	Proposed			Supervised		Full	
	MSE	# of RV	L	MSE	# of RV	MSE	# of RV
abalone	$8.15 \pm 0.79^{**}$	5.40	23.40	9.33 ± 0.85	3.64	4.23	26
airfoil	$13.30 \pm 7.49^{**}$	5.15	27.55	2715.81 ± 994.83	17.06	4.96	122
concrete	$49.17 \pm 40.96^{**}$	3.70	27.00	290.11 ± 108.28	7.71	22.50	63
wine-red	$0.92 \pm 0.22^{**}$	9.05	32.70	1.02 ± 0.13	5.02	0.36	33
Am. option	$19.58 \pm 16.74^{**}$	6.90	46.80	45.88 ± 16.27	7.05	1.27	25

Table 4.4 Experimental results for data sets. *Querying the farthest relevance vector* strategy is used for the querying strategy. # of RV refers to the number of relevance vectors and L is the averaged number of labeled points that the proposed algorithm converges.

Data set	Proposed			Supervised		Full	
	MSE	# of RV	L	MSE	# of RV	MSE	# of RV
abalone	$7.99 \pm 1.24^{*}$	6.60	28.95	8.67 ± 0.91	3.97	4.23	26
airfoil	$4.54 \pm 3.87^{**}$	10.35	85.65	732.78 ± 678.58	35.86	4.96	122
concrete	$75.77 \pm 24.93^{**}$	25.40	177.15	83.21 ± 30.65	22.39	22.50	63
wine-red	$0.73 \pm 0.21^{**}$	11.15	44.45	0.93 ± 0.11	5.70	0.36	33
Am. option	$2.82 \pm 1.62^{**}$	14.55	106.30	22.10 ± 8.14	10.69	1.27	25

similar as new labeled points are added when using the first or last strategy.

The convergence of active learning is sometimes difficult to determine and the active learning algorithm should often be stopped before the convergence. As shown in Figure 4.3(a), the performances of the proposed method near the convergence are not much varied. Table 4.5 shows the average performances of at least 5 results before the convergence. The last querying strategy, *querying the farthest relevance vector*, was exploited for this experiment. This simulation was repeated more than 20 times for each data set and the results were averaged. As a result, MSEs shown in Table 4.5 are not much different from those in Table 4.4 as expected.

Table 4.5 Experimental results for data sets before the convergence. *Querying the farthest relevance vector* strategy is used for the querying strategy. 5 results before the convergence were averaged and these averaged results were averaged again by more than 20 times repeated simulations.

Data set	Proposed		Supervised	
	MSE	# of RV	MSE	# of RV
abalone	8.42 ± 1.70	6.86	8.92 ± 1.00	3.87
airfoil	$4.57 \pm 3.72^{**}$	10.61	800.74 ± 784.14	35.23
concrete	$75.34 \pm 25.53^{**}$	25.53	84.44 ± 32.02	22.16
wine-red	$0.76 \pm 0.22^{**}$	11.23	0.95 ± 0.12	5.62
Am. option	$2.79 \pm 1.61^{**}$	14.58	22.79 ± 8.63	10.58

In summary, the first and third strategies typically resulted in smaller MSEs than the second strategy; however, the latter required the smallest number of labeled points, which can minimize the labeling cost. Meanwhile, the first

strategy queries several points at once, the required number of iterations can be smaller than the others if the converged number of labeled points are similar. Therefore, we recommend the user to select the querying strategy based on the purpose of the task and cost of labeling.

4.4 Chapter Summary

In this study, the novel active learning method for transductive sparse Bayesian regression is proposed. First, we develop a transductive and generalized version of RVM in which the relevance vectors of the constructed model are selected from the unlabeled data points as well as the labeled data points. Next, we propose three querying strategies for the active selection of the labeled point set using only the relevance vectors automatically obtained from the developed model, thereby making an additional process for active learning unnecessary. The proposed active learning algorithm is completed by repeating the two previously-mentioned procedures until the model converges or one of the stopping criteria is satisfied.

The proposed method outperformed the random selection algorithm for both artificial and real data sets, whereas it did not perform well compared with the full RVM model that used the whole data points as labeled. The three strategies showed different characteristics when applied to the real data sets. The first querying strategy, *querying all unlabeled relevance vectors*, and the last querying strategy, *querying the farthest relevance vector*, showed significantly small MSEs and standard errors but they required a large number of

labeled points to converge. The second strategy, *querying the most uncertain relevance vector*, converged with a small number of labeled points, which means that the labeling cost can be minimized, while MSEs and standard errors from this strategy were higher than those of the other strategies.

There are possible ways to improve the proposed method. First, the user should select the initial set of labeled points. In this paper, we randomly selected three points from given data points and the method worked successfully. However, other sophisticated sampling methods, including clustering approaches, may improve the performances of the first few iterations. The stopping criterion is another issue that the user must decide on. It is difficult to predict when the model converges, however, we observed that the proposed methods performed significantly better than the random selection after several iterations, and the MSEs at a few iterations before convergence were comparable to that at convergence. Thus, it seems that an earlier stopping than the optimal model may have prevented the occurrence of severe problems. Finally, other querying strategies which use the information criteria can be developed. Similar to the suggestions of previous studies(Zhang & Oles, 2000; Schein & Ungar, 2007; Settles & Craven, 2008), a querying strategy based on information theory can be applied to the transductive GRVM developed in this study in order to construct another active learning algorithm for sparse Bayesian regression.

Chapter 5

Applications to Financial Technologies

5.1 Chapter Overview

The financial variable prediction has been a long and yet active research theme targeted by many researchers since successful prediction helps to make profits as well as avoid risks. There have been many machine learning approaches for financial markets but there are still remaining the unsolved problems that the learning models can make better results. In this chapter, we applied machine learning models, including both sparse and non-sparse ones, to two financial technology problems: high-frequency market impact costs estimation and credit default swap (CDS) prediction.

The remainder of this chapter is organized as follows. In the next section, we briefly review the machine learning algorithms that are applied to the financial problems. Then, the experimental results of each financial problem are following. First, in section 3, learning models are applied to estimate and predict the market impact costs of US markets, then learning models are used to price CDS spreads in section 4. Finally, we conclude this chapter with summary and some

other future research directions in section 5.

5.2 Preliminaries

Before introducing financial application problems targeted in this chapter, we describe some machine learning algorithms which were commonly used for the following problems. In this section, we review the four learning models: artificial neural networks (ANNs), BNNs, GPs, SVMs, and RVMs. The last two models are sparse models, while the other are not.

5.2.1 Artificial Neural Networks

ANNs(Rosenblatt, 1961) are extensively used highly nonlinear nonparametric model which can be used for the regression task. Mimicking a human brain, an ANN model is composed of layers which also consist of nodes, conducting a role as neurons in the brain. There are usually three types of layers in an ANN: input layer, output layer, and hidden layer. The input layer is the first layer and it has nodes that propagates the value of input variables to the next layer. The output layer is the last layer has nodes that make the overall outputs. The hidden layers are located at between the input layer and the output layer and they have nodes that makes the nonlinear output from inputs propagated from the previous layer. The nonlinear output $f(\mathbf{x})$ of each node has a form as follows:

$$f(\mathbf{x}) = g(\sum_i w_i h_i(\mathbf{x})) \tag{5.1}$$

where $\mathbf{x}$ is the input vector from the previous layer, g is an activation function, h_i's are functions that transform the input vector. The sigmoid functions, S-shaped functions such as hyperbolic tangent function, logistic function, and probit function, are commonly used for the activation function.

Training of ANN means the optimization of weights w_i's in Eq. (5.1). The widely used optimization algorithm is *back-propagation algorithm*(Rumelhart et al., 1985). In back-propagation algorithm, the weights are chosen backwardly from the output layer to the first hidden layer to minimize the loss, squared sum of errors in usual. The ANNs show good performances after training the weights but it is difficult in this model to determine the relationship between inputs and outputs.

5.2.2 Bayesian Neural Networks

BNN is a variant of NN training algorithm, which was originally proposed in MacKay (1992). Similar to other algorithms with Bayesian nature, this algorithm assumes a Gaussian-type prior over weights on the networks, or equivalently regularizes the error function via sum of squares of weights.

Suppose we have n input-output pairs $\{x_i, y_i\}$ with $y_i = f(x_i) + \epsilon_i$, where ϵ_i are i.i.d Gaussian errors. In BNN setting, the objective function is represented as $F(\mathbf{w}, \alpha, \beta; \{x_i, y_i\}) = \beta E_D + \alpha E_W$, where $E_D = \sum_{i=1}^{n} (y_i - a_i)^2$ is the sum of squares of errors from network output a_i and target y_i corresponding to x_i, and E_W is the sum of squares of the network weights. The relative importance between regularization and fitting the data is determined by adjusting relative size between α and β, which can be done by maximizing the posterior distribution

$P(\alpha, \beta|\{x_i, y_i\})$.

Foresee and Hagan (1997) proposed an algorithm which iteratively optimizes the weight and parameters, using Gauss-newton approximation to the Hessian of the objective function F. After initializing α, β and weights, the algorithm runs as follows:

1. Take one step of the Levenberg-Marquardt algorithm to minimize the objective $F(\mathbf{w}) = \beta E_D + \alpha E_W$.

2. Compute the effective number of parameters $\gamma = N - 2\alpha \mathrm{tr}(\mathbf{H})^{-1}$, making use of the Gauss-Newton approximation to the Hessian: $\mathbf{H} = \nabla^2 F(\mathbf{w}) \approx 2\beta J^T J + 2\alpha I_N$ where J is the Jacobian matrix of the training set errors.

3. Compute new estimates for the objective function parameters: $\alpha = \frac{\gamma}{2E_W(\mathbf{w})}$ and $\beta = \frac{n-\gamma}{2E_D(\mathbf{w})}$.

5.2.3 Gaussian Processes

GP regression (Cressie, 1993; Rasmussen, 1996) is a collection of random variables such that any finite combination of them follows the Gaussian distribution. A GP $f(\mathbf{x})$ can be completely determined by its mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ as

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))].$$

Assume that the set of data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ is given with the noisy output y_i where the variance of the output noise is denoted by σ^2. Then, the covariance

of the output vector $\mathbf{y} = (y_1, \ldots, y_n)$ is given as

$$\mathrm{cov}(\mathbf{y}) = \mathbf{K}$$

where $\mathbf{K}$ is an $N \times N$ matrix whose i,j'th entry is $k(\mathbf{x}_i, \mathbf{x}_j)$. Then, with a new input $\mathbf{x}^*$ and define the mean function $m(\mathbf{x}) = 0$,

$$\begin{bmatrix} \mathbf{y} \\ f^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma^2\mathbf{I} & \mathbf{k}_*^T \\ \mathbf{k}_* & k_{**} \end{bmatrix}\right) \tag{5.2}$$

where $f^* = f(\mathbf{x}^*)$, $k_{**} = k(\mathbf{x}^*, \mathbf{x}^*)$, and $\mathbf{k}_* = (k(\mathbf{x}_1, \mathbf{x}^*), \ldots, k(\mathbf{x}_n, \mathbf{x}^*))^T$. Then the distribution for predictive output f^* can be easily calculated by using the conditional distribution for normal distribution as follows:

$$P(f^*|\mathcal{D}) = \mathcal{N}(\mathbf{k}_*^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}, k_{**} - \mathbf{k}_*^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}_*). \tag{5.3}$$

As shown in Eq. (5.3), the GP regression gives the variance of the predictive output as well as the mean value, thus it belongs to the class of Bayesian regression.

Training of GP refers to optimizing the hyperparameters in the kernel function k and the output noise σ^2 by maximizing the log-likelihood function

$$\log P(\mathbf{y}|\mathcal{D}) = -\frac{1}{2}\mathbf{y}^T(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log\det(\mathbf{K} + \sigma^2\mathbf{I}) - \frac{N}{2}\log 2\pi. \tag{5.4}$$

The log-likelihood function in Eq. (5.4) varies with the exploited kernel functions and the most widely used kernel function is a squared exponential kernel function which has the form $k(\mathbf{x}, \mathbf{x}') = C\exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$ where C and γ are hyperparameters. For more details, see Williams and Rasmussen (2006); Bishop (2006).

5.2.4 Support Vector Machines

SVM is originally developed as a binary classifier(Boser et al., 1992; Vapnik, 2000) but has also been used as a regression model(Drucker et al., 1997) extensively. Here, we explain these two versions of SVMs separately as follows.

Support vector machine classification

SVM classifier first utilizes a nonlinear transfer mapping Φ to map all training data into a high-dimensional feature space. Next, to find an optimal linear classifier of the form

$$f(x_i) = w^T \Phi(x_i) + b. \tag{5.5}$$

in the mapped high-dimensional feature space, it tries to find the parameters w and b which make the classifier in Eq. (5.5) optimal in the sense that the margin, the distance between the classifier and the nearest point $\Phi(x_i)$, is maximized. Finding the optimal w and b can be achieved by solving the following optimization problem:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \tag{5.6}$$

subject to $y_i(w^T\Phi(x_i) + b) \geq 1$ where y_i is a binary target of the instance x_i.

In the case of overlapping or misclassified training instances, we can add the penalty term for these misclassification and then the optimization problem in Eq. (5.6) can be changed into:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \tag{5.7}$$

subject to $y_i(w^T\Phi(x_i) + b) \geq 1 - \xi_i$ where ξ_i is a slack variable to allows soft

margins. The solution of Eq. (5.7) is then given by

$$w = \sum_i \alpha_i^* y_i \Phi(x_i) \tag{5.8}$$

where α_i^* is the solution of the following quadratic optimization problem, which is dual of the primal problem (5.7):

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{5.9}$$

Here $k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ is called a kernel function. There are many candidates for the kernel functions and in this paper, the radial basis kernel function is used (Bishop, 2006).

The nearest points x_i that are nearest to the decision boundary are correspondent with $\alpha_i^* > 0$ and are called support vectors. There are two kinds of support vectors. Ones are support vectors for the class +1 and the others are for the class -1. After finding optimal w and b with training data, like the other classifiers, we can classify the test instances as follows:

$$\text{target}_i = \begin{cases} 1 & \text{if } f(x_i) > 0, \\ -1 & \text{otherwise.} \end{cases} \tag{5.10}$$

Support vector machine regression

SVR is a kernel regression that minimizes the ϵ-insensitive loss function

$$\mathcal{L}(y_1, y_2) = \max\{\epsilon, |y_1 - y_2|\} - \epsilon \tag{5.11}$$

with some $\epsilon > 0$. This loss function is zero if $|y_1 - y_2| < \epsilon$ and $|y_1 - y_2| - \epsilon$ otherwise. Defining the regression function $f(\mathbf{x}, \mathbf{w}) =< \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) > +b$ with

basis functions $\boldsymbol{\phi}$ and penalize with the errors larger than ϵ, the SVR problem, which minimizes $\|\mathbf{w}\|^2$ to reduce the complexity of model, becomes

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i^+ + \xi_i^-) \tag{5.12}$$

with the constraints

$$\begin{aligned} y_i - f(\mathbf{x}_i, \mathbf{w}) &\leq \epsilon + \xi_i^+ \\ f(\mathbf{x}_i, \mathbf{w}) - y_i &\leq \epsilon + \xi_i^- \\ \xi_i^+, \xi_i^- &\geq 0 \end{aligned} \tag{5.13}$$

for all $i = 1, \ldots, n$. Using Karush-Kuhn-Tucker conditions, we can get the following dual problem

$$\max_{\boldsymbol{\alpha}^+,\boldsymbol{\alpha}^-} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i^+ - \alpha_i^-)(\alpha_j^+ - \alpha_j^-)k(\mathbf{x}_i, \mathbf{x}_j) - \epsilon\sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-) + \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)y_i \tag{5.14}$$

with the constraints $0 \leq \alpha_i^+, \alpha_i^- \leq C$ where the kernel function, $k(\mathbf{x}_i, \mathbf{x}_j) =< \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j) >$ and $\mathbf{w} = \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)\boldsymbol{\phi}(\mathbf{x}_i)$. This dual problem can be solved by a quadratic programming solver and then the predictive value for the new input $\mathbf{x}^*$ becomes

$$y^* = \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)k(\mathbf{x}_i, \mathbf{x}^*) + b \tag{5.15}$$

where $b = y_k - \sum_{i=1}^{n}(\alpha_i^+ - \alpha_i^-)k(\mathbf{x}_i, \mathbf{x}_k)$ for any $k = 1, \ldots, n$. For more details, see Drucker et al. (1997); Bishop (2006).

5.2.5 Relevance vector machines

RVM, firstly proposed in Tipping (2001), is a Bayesian regression whose weight vector has an ARD prior. Here, we briefly give the formalism of RVM regression.

Like other regression models using basis functions, RVM regression has the form as follows:

$$y = f(\mathbf{x}) + \epsilon \tag{5.16}$$

where $\epsilon \sim \mathcal{N}(0, \beta^{-1})$ and $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})$ where $\mathbf{w}$ is a weight vector and $\boldsymbol{\phi}(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_N(\mathbf{x})]^{\mathrm{T}}$ is a vector of basis functions. For RVM, a kernel function $\kappa(\mathbf{x}, \mathbf{x}')$ is used as a basis function thus the i'th basis function $\phi_i(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}_i)$. $\phi_0(\mathbf{x}) = 1$ is sometimes included in the basis vector to represent the bias term. Also, a weight vector has an ARD prior, $p(\mathbf{w}|\mathbf{A}) \sim \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{A}^{-1}) = \prod_{i=1}^{N} \mathcal{N}(w_i|0, A_{ii}^{-1})$ with the diagonal matrix $\mathbf{A}$ as the hyperparameter.

The two hyperparameters, the noise precision β and the prior precision $\mathbf{A}$, can be found by typical maximum likelihood estimation where the likelihood function is defined as

$$\begin{aligned} p(\mathbf{y}|\mathbf{X}, \mathbf{A}, \beta) &= \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \beta) p(\mathbf{w}|\mathbf{A}) d\mathbf{w} \\ &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^{\mathrm{T}} + \beta^{-1}\mathbf{I}) \end{aligned} \tag{5.17}$$

where $\boldsymbol{\Phi}$ is a $N \times N$ matrix whose i'th row is $\boldsymbol{\phi}(\mathbf{x}_i)^{\mathrm{T}}$. Because of the ARD prior, most of diagonal elements in $\mathbf{A}$ become infinite as a result of likelihood maximization and the corresponding weights become zero. Thus, the regression model in (5.16) can be represented only with a small number of kernel functions relevant to the input points corresponding to the nonzero weights and these points are called *r*elevance vectors. The detailed description for this sparsity can be found in Tipping (2001) and Tipping and Faul (2003). Then the predictive distribution for a new point $\mathbf{x}^*$ are derived as following:

$$p(y^*|\mathbf{x}^*, \mathbf{f}, \mathbf{X}, \mathbf{A}, \beta) \sim \mathcal{N}(\overline{f}(\mathbf{x}^*), \beta^{-1} + \boldsymbol{\phi}(\mathbf{x}^*)^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\phi}(\mathbf{x}^*)) \tag{5.18}$$

where

$$\overline{f}(\mathbf{x}^*) = \beta\boldsymbol{\phi}(\mathbf{x}^*)^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{y} \tag{5.19}$$

$$\boldsymbol{\Sigma} = (\mathbf{A} + \beta\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-\mathbf{1}}. \tag{5.20}$$

5.3 Analyzing market impact costs using nonparametric learning models

5.3.1 Motivation

Transaction costs have been considered as one of the important factors affecting the investment performance for a long time. As statistical and computational technologies have been adapted to estimate and predict several financial variables from a few decades ago, the high-frequency trades of financial assets have been popular and thus the estimation and prediction of transaction costs becomes more important. Transactions costs are usually classified into two major categories: explicit costs and implicit costs. Explicit costs, also called direct costs, are transaction costs that can be explicitly stated and measured. These costs include commissions, transaction fees, and taxes. Implicit costs, or indirect costs, are costs that cannot be measured directly including bid-ask spreads, time risk costs, and market impact costs. These costs are usually regarded improvable by an appropriate trading strategies.

Market impact cost, one of the implicit transaction costs, the cost caused by the difference between the price before the transaction and the price that the transaction is executed actually. There have been several literatures focusing on analyzing market impact costs. Lillo et al. (2003) and Gabaix et al. (2003)

fitted the impacts of single transactions to a concave power-law function of the volume of the transaction. Bouchaud et al. (2004) used a logarithm function of the transaction volume to estimate the market impact costs. Plerou et al. (2002) exploited the hyperbolic tangent function for the same task. Almgren et al. (2005) and Kato (2014) used a stochastic process of the asset price which includes a function of the transaction size to explain the market impacts. Frino et al. (2008) estimated the impact cost by using linear regression with quantized transaction sizes. Bershova and Rakhlin (2013) analyzed the market impacts of the large institutional orders in the US equity market and found that the permanent impact function has a concave form with respect to the transaction size in contrast to the other previous results(Almgren et al., 2005; Huberman & Stanzl, 2005) that the permanent impact function has a linear form. There have been some other researches using other input variables to estimate the market impact costs. I-star model described in Kissell et al. (2003) and Kissell (2013) is a log-linear regression model which uses three inputs, transaction size, volatility, and underlying trading rate and they affected the estimated market impact costs independently. Bikker et al. (2007) and Bikker et al. (2008) used more than 40 independent variables to fit the market impact cost to simple linear regression function. However, most of these previous studies showed the limitation in performance because of the fixed parametric or simple linear regression form of the market impact model.

Nonparametric machine learning models have been preferred to be applied to various other areas including financial data analysis due to their abilities in fitting and predicting performances for complex data sets. Most of those finan-

cial applications have been focused on the stock price prediction(W.-H. Chen et al., 2006; Son et al., 2012; Ticknor, 2013; Liao & Chou, 2013) and its derivative markets(Hutchinson et al., 1994; Han & Lee, 2008; H. Park et al., 2014) and most of them showed the accurate prediction results. There have also been several studies for other markets including credit and its derivative markets(Y.-C. Lee, 2007; Gündüz & Uhrig-Homburg, 2011; K.-j. Kim & Ahn, 2012), fixed-income markets(S. H. Kim & Noh, 1997; Cao & Tay, 2003), and foreign exchange markets(Bhattacharyya et al., 2002). Although nonparametric models have been successfully applied to diverse financial applications, they have not been employed to analyze the market impact costs yet.

In this section, we introduce nonparametric approaches to estimate and predict the market impact costs. To the best of our knowledge, this is the first approach which applies nonparametric learning models to analyze the market impact cost. The proposed nonparametric approach has two main advantages. First, the nonparametric approaches usually fit the data better than the parametric case. Second, the nonparametric approaches have a versatility in the number of input variables so the general procedure does not change when the number or kinds or input variables change while the parametric approaches require the new parametric models in those cases. Trough simulation, we analyzed the market impact costs of transactions of small-cap, mid-cap, and large-cap stocks in US equity market both altogether and separately by selecting the same types of input variables with I-star model(Kissell, 2013; Kissell et al., 2003) and compared the results.

5.3.2 High-frequency Trade with Transaction Costs

Son et al. (2012) applied four learning models, linear regression, logistic regression, NN, and SVM, to predict the trend of high-frequency Korea Stock Price Index (KOSPI) 200, by using the market lead-lag relationship. The brief results are shown in Table 5.1. ANN refers to the NN model and SVM refers to the

Table 5.1 Prediction accuracy of four classifiers

Classifier	Linear reg.	Logistic reg.	ANN	SVM
Dataset 1	0.624	0.612	0.612	0.611
Dataset 2	0.622	0.625	0.635	0.628
Dataset 3	0.611	0.613	0.618	0.616
Average	0.619	0.616	0.621	0.618

SVMs. Dataset 1 contains high-frequency data from 2011 Mar 7th to 11th as a training set and data from 2011 Mar 14th to 18th as a test set. As rolling-over the datasets, dataset 2 contains data from 2011 Mar 14th to 18th as a training set and data from 2011 Mar 21st to 25th as a test set. Dataset 3 has the training data from 2011 Mar 21st to 25th and the test data from 2011 Mar 28th to Apr 1st. This rolling-over of datasets is presented in Figure 5.1.

Table 5.2 Virtual trading returns (in percentage)

Classifier	Base	Linear reg.	Logistic reg.	ANN	SVM
Dataset 1	6.28	79.44	75.70	73.52	69.92
Dataset 2	3.54	31.80	31.93	35.47	33.67
Dataset 3	6.88	29.42	29.89	30.85	29.80
Average	5.57	46.88	45.84	46.61	44.46

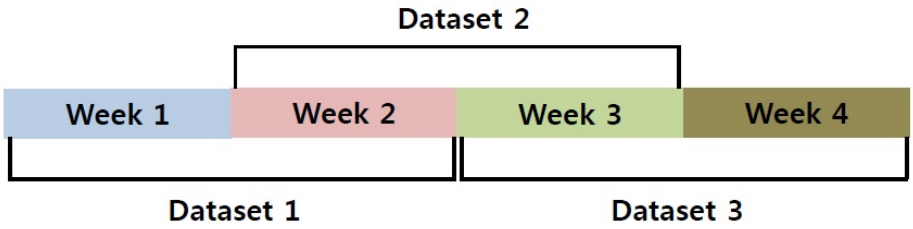


Figure 5.1 The dataset separation with rolling-over.

The virtual trading based on these learning models results in the returns shown in Table 5.2. These returns from the learning models are much higher than the baseline, a simple buy-and-hold strategy. However, these results in Table 5.2 did not consider the transaction cost. Since the trades in Son et al. (2012) were conducted every one minute, the transaction cost can affect the result significantly. For example, if transaction cost for each transaction is 0.3%, which is the transaction tax rate of Korean stock market, the cost becomes about 506% per a week even with neglecting the compound calculation. Therefore, predicting transaction costs at a certain market situation is important for the maximization of the returns especially for the high-frequency trade using the developed technologies.

5.3.3 Review of I-star model

In this section, we briefly review one benchmark parametric model, I-star model(Kissell, 2013; Kissell et al., 2003), which uses three input variables to describe the market impact cost. The I-star model is composed of two separated equations calculating I^*, a theoretical instantaneous cost, and MI, the market impact cost

appeared in the real market, respectively. The equations calculate them are given as follows:

$$I^* = a_1 \cdot Size^{a_2} \cdot Vol^{a_3} \tag{5.21}$$

$$MI = b_1 I^* \cdot POV^{a_4} + (1 - b_1) I^* \tag{5.22}$$

where $Size$, Vol, and POV are input variables and a_1, a_2, a_3, a_4, and b_1 are parameters to be determined.

The first input variable of (5.21) and (5.22) is $Size$, the normalized order size. According to Kissell (2013), it is represented as $Size = Q/ADV$, where Q is the imbalance, the absolute value of difference between buy order and sell order, and ADV is 30-day average daily volume. Thus $Size$ implies the magnitude of pressure from this order relative to the averaged daily volume. The second input variable, Vol, is the volatility of the equity return and 30-day averaged volatility was used in Kissell (2013). The last input variable, POV, is an acronym for *percentage of volume* and it reflects the market liquidity condition. Kissell (2013) simply expressed $POV = Q/(Q + V)$ where V is the expected volume traded for the period of time that the imbalance order Q is executed. If the market is liquid or the imbalance trade order Q is executed slowly, V becomes large and thus POV becomes small. Small POV results in small MI value so the market impact cost will be small when the market is liquid.

The market impact cost in (5.22) is composed of two components, temporary impact cost and permanent impact cost which are the first and last term in the right hand side of (5.22) respectively. Since $Size$ and Vol are used to calculate

the value of I^*, they affect both the temporary and permanent part of the market impact. However, the other input variable POV only appears in the temporary impact part. This implies that the smaller POV, when the other input variables are invariant, incurs the smaller market impact cost but this effect is temporarily and the permanent impact to the market is independent of the market liquidity condition.

There are several parameters that should be estimated. These parameters can be determined with data sets, including input variable values and market impact costs observed in the market, by general parameter estimation techniques such as nonlinear optimization and grid search.

5.3.4 Data Description and Procedures

We describe the proposed procedure to calculate the market impact cost by using nonparametric regression models with the example of single transaction data of representative US stocks. The proposed nonparametric approaches can also separate permanent and temporary costs with an appropriate selection of input variables.

General procedures

First, we mention the general procedure to find market impact costs by using nonparametric regression models before the descriptions of the simulation conducted in the current paper. The whole procedure is classified into three stages: data collection, data preprocessing, and cost analysis. Figure 5.2 represents the summary of the whole procedure.

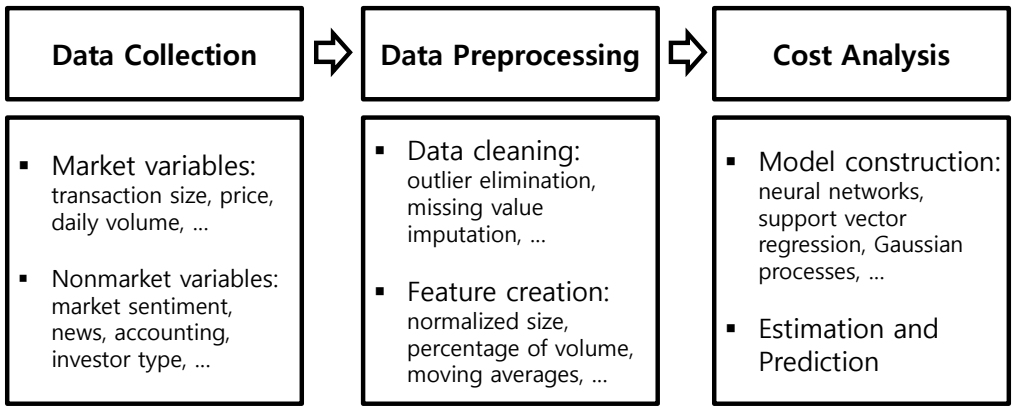


Figure 5.2 Summary of the general procedure of nonparametric approach for market impact cost.

The main task at the first stage, data collection, is to gather necessary data. Collecting non-traditional data outside of the market like news, reports, opinions, and any other variables than may affect the price or liquidity can also be useful as well as the traditional market variables because the nonparametric models do not require any restriction on the data and the general procedure of analyzing market impact costs using them will not be changed.

The gathered data at the first stage are preprocessed to make input variables at the second stage. First, data cleaning processes like outlier elimination and missing value imputation are conducted. Then, the input variables which will be used for the nonparametric models are derived from these cleaned data.

At the final stage, nonparametric models to estimate and predict market impact costs are constructed using input variables created in the previous stage.

Using the constructed models, the diverse analysis of data-driven market impact costs such as determining and permanent and temporary portion can be conducted.

Data description

For the simulation of the proposed nonparametric approach, we gathered the single transaction data of the stocks of US equity markets from Bloomberg terminal for the period from 2014/06/02 to 2014/06/26. We selected 17 representative firms which have large market capitals among each of S&P 500, S&P MidCap 400, and S&Ps SmallCap 600 indices for large cap, mid cap, and small cap firms respectively. The tickers of the selected firms are presented in Table 5.3.

The collected transactions data are classified into three data sets, *large cap*, *mid cap*, and *small cap* by their capitals and another data set *all cap* which includes all transactions regardless of the market capital. For each size of capitals, the number of collected transactions are about 15 million, 2 million and 1 million for *large cap*, *mid cap*, and *small cap*, respectively and thus *all cap* data set has about 18 million transactions in total. The procedures in the following sections will be applied commonly to all of those data sets.

Creating and bucketing input variables

We made three input variables, $Size$, Vol, and POV, which correspond to I-star model(Kissell et al., 2003; Kissell, 2013) and one output variable, the market impact cost. Since the I-star was originally applied to the daily-aggregated

Table 5.3 Tickers of selected firms. 17 firms having large market capitals among each of large, mid, and small cap indices by S&P are chosen.

Large cap	*Mid cap*	*Small cap*
AAPL	ADS	FNGN
XOM	AMG	TDY
GOOGL	GMCR	WST
GOOG	TSCO	DAR
MSFT	MHK	WWW
JNJ	LKQ	TYL
WFC	HFC	TTC
GE	HSIC	CGNX
CVX	DDD	QCOR
WMT	PII	CNC
JPM	UA	ENS
PG	CHD	MDSO
VZ	BEAV	LHO
IBM	XEC	VSAT
PFE	JBHT	MMS
T	TRMB	VDC
ORCL	EQIX	SF

transactions, we slightly modified the input variables suitable for the single high-frequent transactions. First, we define the market impact cost, denoted by *cost*, as

$$cost = side \cdot \log(p_t/p_0) \cdot 10^4 \tag{5.23}$$

where *side* is 1 if a trade is a buy-initiated trade and -1 if a trade is a sell-initiated trade, p_0 is a mid-price just before the trade, and p_t is an executed

price of the trade. Since *cost* is multiplied by 10^4, the unit of *cost* becomes basis point (bp). The first input variable $Size$ is the normalized trade size as follows:

$$Size = \frac{V_t}{ATV} \tag{5.24}$$

where ATV is the average trade volume of the previous day. In the original I-star model, the imbalanced trade size is normalized by 30-day average daily volume since the trade size itself is daily=aggregated. Thus, to apply the single transactions, we divide each trade size by the average single trade size of the previous day. The second input variable Vol is defined as the 30-day volatility and this is the same with the original I-star model since the volatility is the characteristic of each stock, not related to the trade size or frequency. POV, the percentage of volume, in Kissell (2013) is defined as Q/V where Q is the daily imbalanced size and V is the total trade volume of that day. A single transaction may be affected by the market liquidity more locally rather than the liquidity of the whole day. Thus we define POV for single transactions as

$$POV = \frac{V_t}{V_t(-\tau, \tau)} \tag{5.25}$$

where $V_t(-\tau, \tau)$ is the total traded volume from τ minutes before the trade to τ minutes after the trade. According to the previous study(Frino et al., 2008), we expected that the single transaction affects and is affected by the market within about 15 minutes and thus we decided τ equals to 15.

After creating input variables, we made three dimensional bins of input and bucketed the transactions into them. For each bin, $Size$ has the value of multiples of 0.01, i.e. $0, 0.01, 0.02, \ldots$, and Vol has the value of multiple of 0.05.

POV has the values of multiples of 0.0002 for *large cap* data set and multiples of 0.001 for the other data sets. Each transaction was bucketed to the bin which has the nearest value. For example, a transaction from *mid cap* data set with the input variables $(Size, Vol, POV) = (0.0137, 0.022, 0.0038)$ was put to the bin having the values $(Size, Vol, POV) = (0.01, 0.02, 0.004)$. The output, *cost*, of each bin is defined by the average *cost* of transactions belonging to the bin.

Finally, we selected bins containing enough number of transactions. The criterion number will be different for data sets. We selected the bins containing more than 20, 30, 60, 100 transactions and then the number of survived bins are 2931, 3356, 5706, 5119 for *small cap*, *mid cap*, *large cap*, and *all cap*. If the selecting criterion with 100 transactions, there remain 2721, 1627, and 1106 buckets for *large cap*, *mid cap*, and *small cap*, respectively. The sum of those buckets is 5454 and it does not exceed the number of selected buckets from the all transactions, 5119. Therefore, we can expect that the selected buckets from *all cap* data are mostly made up with the buckets from transactions of each capital size group.

Analyzing market impact costs

To the nonparametric machine learning models to the bins of transactions, we set 70% of survived bins as the training set the rest of 30% as the test set for each data set. To find appropriate parameter sets of nonparametric models, we used 10-fold cross validation for the training set. After finding the parameter set, each model was retrained for the whole training set with the chosen parameter set and applied to the test set. As a parametric benchmark, we used I-star

model with the same data sets. As described in section 5.3.3, I-star model also requires to find some parameters. We found the parameters for I-star model by grid search and 10-fold cross validation of the training set and applied it to the test set as the same with the nonparametric models.

5.3.5 Simulation results

First, we applied the nonparametric machine learning models and the benchmark parametric model, I-star model, to the selected bins of each data set. We used four different measures, mean absolute error (MAE), relative MAE (RMAE), root mean squared error (RMS), and relative RMS (RRMS), to estimate the errors of the model. The summarized results are shown in Table 5.4 through 5.7. *NN*, *BNN*, *SVR*, *GP*, *RVM* and *I-star* refer to results of NN, BNN, SVR, GP, RVM, and I-star model, respectively.

Through Table 5.4 to 5.7, we can notice that the nonparametric approach fits the data distribution better than the parametric benchmark with the same input features and instances as we expected. However, the performances of nonparametric models were also different among the models. For example, BNNs reduced the errors from 7.27% to 43.00% relative to I-star model but SVR reduced the errors just from -0.005% to 15.03%. This phenomenon is more clarified by Figure 5.3 which represented the errors in the tables above.

We can easily find that the four nonparametric models, NN, BNN, GP, and RVM show much better performances than the parametric benchmark while SVR model performs slightly better than the benchmark and worse than the other nonparametric models in general. In some cases like RMS for *small cap*

Table 5.4 Test errors of the nonparametric models and the parametric benchmark models for *small cap* data set. Cross validation errors are also displayed in the parentheses. The best model for each error measure is **boldfaced**.

Methods	MAE	RMAE	RMS	RRMS
NN	0.9445 (0.9910)	0.3006 (0.3175)	1.4945 (1.5305)	0.4535 (0.4990)
BNN	0.9310 (1.0025)	0.3023 (0.3204)	**1.4559** (1.5286)	0.4502 (0.4820)
GP	**0.8794** (0.8701)	**0.2854** (0.2716)	1.4945 (1.3950)	**0.4442** (0.4060)
SVR	1.0121 (1.0333)	0.3352 (0.3373)	1.5783 (1.5762)	0.5090 (0.5340)
RVM	0.9732 (1.0188)	0.3305 (0.3414)	1.4667 (1.5512)	0.4968 (0.5248)
I-star	1.0396 (1.0446)	0.3410 (0.3408)	1.5701 (1.5891)	0.5097 (0.5476)

Table 5.5 Test errors of the nonparametric models and the parametric benchmark models for *mid cap* data set. Cross validation errors are also displayed in the parentheses. The best model for each error measure is **boldfaced**.

Methods	MAE	RMAE	RMS	RRMS
NN	**0.5266** (0.5254)	0.2831 (0.2932)	0.7851 (0.7542)	0.4184 (0.4381)
BNN	0.5405 (0.5186)	0.2914 (0.2889)	0.7892 (0.7423)	0.4188 (0.4338)
GP	0.5517 (0.5178)	**0.2802** (0.2778)	0.8311 (0.7597)	**0.3907** (0.4144)
SVR	0.6202 (0.5936)	0.3268 (0.3251)	0.8914 (0.8358)	0.4672 (0.4746)
RVM	0.5524 (0.5340)	0.3180 (0.3185)	**0.7815** (0.7442)	0.4677 (0.4843)
I-star	0.6540 (0.6226)	0.3453 (0.3424)	0.9373 (0.8730)	0.4972 (0.5080)

Table 5.6 Test errors of the nonparametric models and the parametric benchmark models for *large cap* data set. Cross validation errors are also displayed in the parentheses. The best model for each error measure is **boldfaced**.

Methods	MAE	RMAE	RMS	RRMS
NN	0.1287 (0.1283)	0.1515 (0.1506)	0.1732 (0.1738)	**0.2051** (0.2054)
BNN	0.1267 (0.1280)	**0.1502** (0.1505)	0.1712 (0.1735)	0.2066 (0.2061)
GP	0.1338 (0.1377)	0.1583 (0.1621)	0.1802 (0.1878)	0.2172 (0.2123)
SVR	0.1872 (0.1896)	0.2267 (0.2300)	0.2466 (0.2459)	0.3085 (0.3112)
RVM	**0.1255** (0.1292)	0.1520 (0.1558)	**0.1709** (0.1738)	0.2156 (0.2179)
I-star	0.2203 (0.2229)	0.2635 (0.2661)	0.2823 (0.2823)	0.3484 (0.3503)

Table 5.7 Test errors of the nonparametric models and the parametric benchmark models for *all cap* data set. Cross validation errors are also displayed in the parentheses. The best model for each error measure is **boldfaced**.

Methods	MAP	RMAP	RMS	RRMS
NN	0.4096 (0.3746)	0.4096 (0.2173)	0.7557 (0.6388)	0.3507 (0.3210)
BNN	**0.3789** (0.3683)	**0.2182**(0.2170)	**0.6667**(0.6251)	**0.3192** (0.3292)
GP	0.4327 (0.4059)	0.2586 (0.2519)	0.7383 (0.6598)	0.3601 (0.3576)
SVR	0.4488 (0.4256)	0.2766 (0.2710)	0.7485 (0.6840)	0.3933 (0.3964)
RVM	0.3914 (0.3853)	0.2357 (0.2383)	0.6965 (0.6412)	0.3541 (0.3681)
I-star	0.4747 (0.4517)	0.2989 (0.2931)	0.7784 (0.7029)	0.4163 (0.4149)

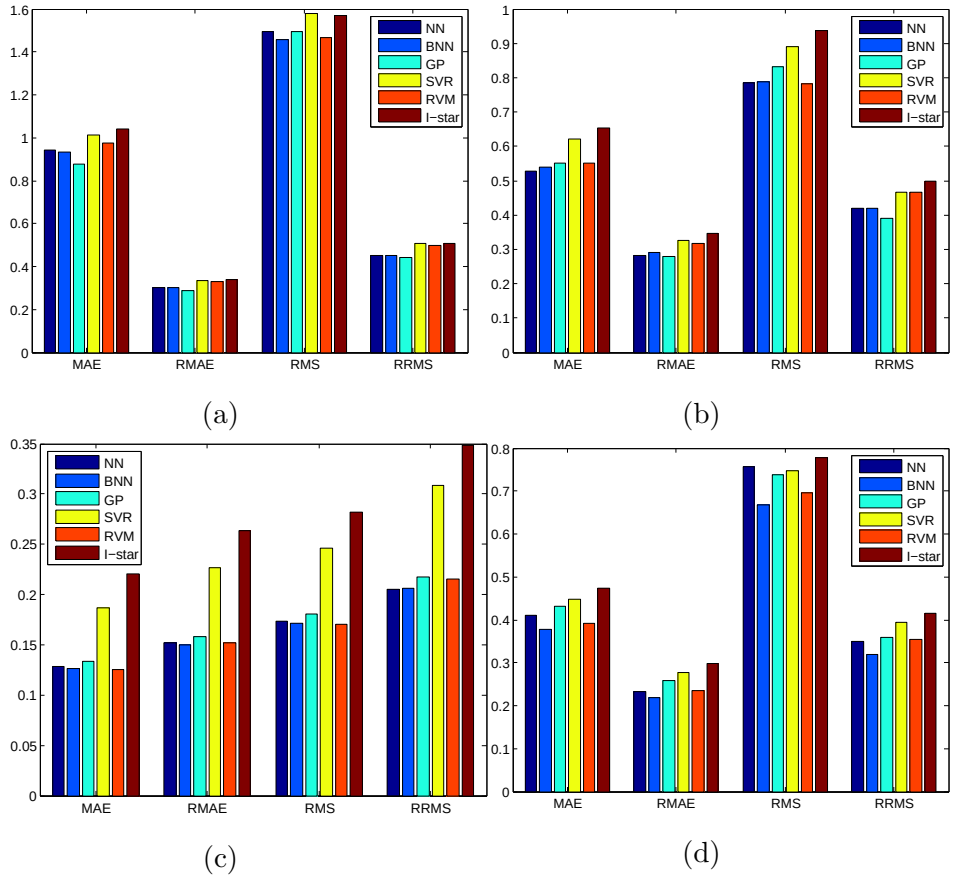


Figure 5.3 Test errors of the nonparametric machine learning models and the parametric benchmark. (a) *small cap* data set (b) *mid cap* data set (c) *large cap* data set (d) *all cap* data set

data set, SVR performs even worse than the benchmark model.

Table 5.8 Training and test time for GP and RVM. A unit for all values is second.

	GP		RVM	
	Training time	Test time	Training time	Test time
small cap	49.51	0.2321	2.1956	0.0263
mid cap	20.16	0.3112	0.6857	0.0368
large cap	65.46	0.4302	0.5840	0.0449
all cap	549.33	0.8611	7.5780	0.0651

RVM and GP have their Bayesian property in common, but the former is sparse whileas the latter is not. The main advantage of sparse models is fast computation compared to the non sparse models maintaining the comparable performances. Table 5.8 shows the training and test time for two methods. It is noticeable that both training and test time of RVM are much faster than those of GP while those models showed comparable performances as represented through Table 5.4 to Table 5.7, this is a typical result from the sparse and non sparse models.

For RVM, we applied the active learning algorithm proposed in chapter 4 to the training set of each data set with the parameters selected by 10-fold cross validation above and the results are shown in Table. 5.9. *Q1*, *Q2*, and *Q3* refer to the first, second, and third querying strategies in section 4.3.2, respectively.

The results coincide the results in chapter 4. Active learning algorithms except the second querying strategy showed better MSEs than the random

Table 5.9 Active learning results for data sets. Three querying strategies in section 4.3.2 are denoted by *Q1*, *Q2*, and *Q3*. # of RV refers to the number of relevance vectors and L is the averaged number of labeled points that the proposed algorithm converges.

Data set and		Active			Random selection		Full		
Querying strategy		MSE	# of RV	L	MSE	# of RV	MSE	# of RV	L_{total}
small cap	*QS1*	2.6358 ± 0.2126	46.83	246.50	3.0467 ± 0.0891	36.38	1.8484	69	2052
	QS2	3.4432 ± 0.1914	32.00	192.17	3.4906 ± 0.2351	33.67			
	QS3	2.3277 ± 0.0256	52.00	265.83	3.0690 ± 0.0662	37.70			
mid cap	*QS1*	0.8083 ± 0.0647	11.00	78.67	0.8886 ± 0.0533	10.30	0.5187	24	2350
	QS2	0.9124 ± 0.1645	10.50	63.00	1.0060 ± 0.1372	9.42			
	QS3	0.7108 ± 0.0328	13.17	87.67	0.8436 ± 0.0211	10.70			
large cap	*QS1*	0.0377 ± 0.0027	29.17	178.67	0.0471 ± 0.0036	24.60	0.0273	43	2782
	QS2	0.0420 ± 0.0028	31.17	213.33	0.0443 ± 0.0009	26.03			
	QS3	0.0381 ± 0.0011	31.83	206.67	0.0462 ± 0.0026	25.23			
all cap	*QS1*	0.4781 ± 0.0189	62.17	420.00	0.6324 ± 0.0255	54.70	0.3098	102	3584
	QS2	0.6685 ± 0.0575	50.50	291.33	0.7257 ± 0.0472	48.53			
	QS3	0.4294 ± 0.0225	67.17	472.33	0.6048 ± 0.0357	56.82			

selection, while using the whole data set as labeled showed the best performance. The second querying strategy performs slightly better than the random selection in the sense of mean MSEs but these results were not statistically significant, while the results of the other two strategies were statistically significant with the level of confidence $\alpha = 0.01$. The number of relevance vectors for the results of Active learning is much lower than those of the full RVM results and a slightly higher than those of the results of random selections. The number of labeled points, of course, were much less than the total number of input points.

5.4 Predicting Credit Default Swaps via Nonparametric Learning Models

5.4.1 Motivation

Credit market is one of the most important financial markets and has received wide attention especially from the credit crisis in 2008. A default probability is one of the typical measures to represent the credit risk of a firm or nation but it is difficult to determine since many firms and nations, or obligors, are linked by various contracts and obligations and thus a credit-related event, including default, of one obligor may affects many other obligors. The default probability is usually measured by the credit derivatives traded in the credit market because their prices do not highly affected by the other factors than credit risk unlike defaultable bond prices. For example, Bühler and Trapp (2009) showed that the 95% of the credit default swap (CDS) spread stems from the credit risk while only 4% from the liquidity. The mispricing of these derivatives can lead

to misunderstanding of default probability. Also, credit estimation becomes the core part of individual lending market which is one of famous areas of financial technologies. Thus, accurate pricing for credit derivatives from the credit crisis period has become an important consideration.

During the last two decades, many researches have been made to price credit derivatives and their models can be categorized into two classes of models. One class of models, called *structural models*, assume that a certain stochastic process for the fundamental value of the firm and defines an event of default as the fundamental value hits a predetermined barrier(Merton, 1974; Black & Cox, 1976; Finger et al., 2002). the other class of models, called *reduced-form models* or *intensity-based models*, assume that the default is driven by an exogenous factors and an event of default follows a Poisson process with a stochastic intensity(Vasicek, 1977; J. C. Cox et al., 1985; Jarrow & Turnbull, 1995). There have also existed a large number of studies that compared those models by the predicted credit derivative prices(Jones et al., 1984; Ogden, 1987; Duffee, 1999; Lyden & Saraniti, 2001; Eom et al., 2004; Bakshi et al., 2006; Gündüz & Uhrig-Homburg, 2011) but there has not been a robust conclusion that a certain model overwhelms the others for pricing and predicting credit derivatives traded in the real market.

On the other hand, nonparametric learning models have extensively been used to predict financial time series in recent years due to their flexibility which fits the models to the data well. Most of those results have been focused on the stock(W.-H. Chen et al., 2006; Son et al., 2012; Ticknor, 2013; Liao & Chou, 2013) and its derivative markets(Hutchinson et al., 1994; Han & Lee,

2008; S.-H. Yang & Lee, 2011; H. Park & Lee, 2012; H. Park et al., 2014) and achieved accurate prediction results. However, relatively a few studies have been conducted for the other markets including the fixed-income market(S. H. Kim & Noh, 1997; Cao & Tay, 2003) and the foreign exchange market(Bhattacharyya et al., 2002). For the credit market, most of studies using learning methods have concentrated on credit rating analysis. Y.-C. Lee (2007) and K.-j. Kim and Ahn (2012) used SVMs to classify the rating of firm and Z. Huang et al. (2004) classified the rating of corporate bonds using both SVMs and ANNs. For credit derivatives pricing, Gündüz and Uhrig-Homburg (2011) applied the SVR (Drucker et al., 1997) to predicting one-dimensional output corresponding five-year maturity CDS spread of one firm using the spreads of other firms at the same moment called a cross-sectional design or using the past value of spreads of the same firm called a time series design and compared it with those of the Merton model (Merton, 1974) and the constant intensity model (Jarrow & Turnbull, 1995). However, only one specific spread was used in prediction although considering and predicting the spreads of other liquid maturities at the same time are practically important and no advanced state-of-the-art machine learning models other than SVR were used for comparison.

To our knowledge, no empirical studies have been made that prices and predict the multi-valued CDS spreads using several nonparametric learning models and even the earlier studies on other financial markets such as stocks or options were made to determine and predict one-dimensional outputs for its price values. In this chapter we aim to conduct a comprehensive study that compares the predictive power of several nonparametric models using the

multi-valued real CDS spread data from January 2001 to February 2014 as well as those of different credit ratings. For our experiment, we applied four well-known state-of arts nonparametric learning regression models (*support vector regression*(SVR)(Drucker et al., 1997), *artificial neural networks*(ANNs), *Bayesian neural networks*(BNN), *Gaussian processes*(GPs) (Cressie, 1993; Rasmussen, 1996)), and *relevance vector machine*(RVM)(Tipping, 2001) to prediction of six dimensional outputs consisting of CDS spreads with six different maturities, 1, 2, 3, 5, 7, and 10 years. Also to verify the relative predictive performance of nonparametric learning models, we've applied a benchmark parametric model, called constant intensity model (Jarrow & Turnbull, 1995) that showed a better result than other parametric models consistently.

5.4.2 Structure of CDS

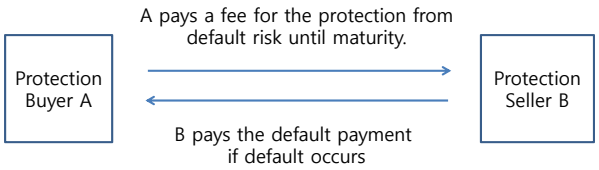


Figure 5.4 The structure of CDS.

A simple structure of CDS (Schönbucher, 2003) is described in Figure 5.4 as a preliminary. Assume that a party A has a risky asset and wants to protect itself from the default risk. If another party B wants to ensure A this protection,

they may agree a CDS contract. By the CDS contract, A pays a fixed fee, called *premium leg*, periodically until the default occurs or the contract matures and B pays default payment, called *protection leg*, to A when the default occurs. The amount of default payment is usually the same with the loss of A from default. As a result, A gets a protection from a default risk while it maintains the returns from the risk asset and B gets a fixed fee periodically.

Pricing the CDS spreads belongs to the multi-valued regression problem since its output consists of several continuous variables, each of which corresponds to a spread of one maturity with the same type. To price and predict the CDS spreads, we next review one benchmark parametric model, called constant intensity model (Jarrow & Turnbull, 1995) that showed a relatively better result than other parametric models consistently and four state-of-the-art machine learning models that are known to have high flexibility and predictive power: ANNs, BNNs, SVR, and GP regression. Each of these models is briefly explained below.

5.4.3 Parametric Constant intensity model

Constant intensity model(Jarrow & Turnbull, 1995), used as a benchmark in this section, is one of the widely used reduced-form models for credit derivative pricing. Reduced-form models generally assume that the default happens stochastically and independently from market information thus they do not assume any fundamental values unlike the structural models which assume a fundamental asset value process of a firm. The process of default in the reduced-form models is usually described as a Cox process(D. R. Cox, 1955), which is

a Poisson process with a stochastic intensity. The intensity, or hazard rate, of default process is defined as

$$\lambda_t = \lim_{\delta t \to 0} \frac{P(t < \tau < t + \delta t)}{P(\tau > t)\delta t} \tag{5.26}$$

where $P(\tau \in B)$ denotes the probability that the default time τ is included in a set of time B. Once the intensity process is defined, the default probability density $P(\tau \in [t, t + dt])$ can be easily calculated as

$$P(\tau \in [t, t + dt]) = \mathbb{E}[\lambda_t e^{-\int_0^t \lambda_s ds}]dt. \tag{5.27}$$

In constant intensity model, the intensity process is defined as a constant, i.e. $\lambda_t = \lambda$. Thus, the Cox process for the default becomes a Poisson process with intensity λ and so the default time follows the exponential distribution with the parameter λ. Thus the survival probability in this model simply becomes $P(\tau > t) = e^{-\lambda t}$ and the premium leg and protection leg become as follows(Duffie & Singleton, 2012):

$$V_{\text{premium}} = F\hat{s}\sum_{i=1}^{N} e^{-(r(i)+\lambda)T(i)}(T(i) - T(i-1)) \tag{5.28}$$

$$V_{\text{protection}} = F(1-R)\sum_{i=1}^{N} e^{r(i)T(i)}(e^{-\lambda T(i-1)} - e^{-\lambda T(i)}). \tag{5.29}$$

Equating two values above, we obtain the fair CDS spread in this model:

$$\hat{s} = \frac{(1-R)\sum_{i=1}^{N} e^{r(i)T(i)}(e^{-\lambda T(i-1)} - e^{-\lambda T(i)})}{\sum_{i=1}^{N} e^{-(r(i)+\lambda)T(i)}(T(i) - T(i-1))}. \tag{5.30}$$

and simply $\hat{s} = \frac{(1-R)(e^{\lambda \Delta t}-1)}{\Delta t}$ when the time interval $T(i+1) - T(i) = \Delta t$ for all $i = 1, \ldots, N$.

5.4.4 Design of experiments

Data description

We used daily CDS contract data obtained from *MARKIT* database. The whole period of data set used is from January 2001 to February 2014. First, we eliminated data by the type of currency and region. The only US dollar denominated and Northern American contracts were used for the experiments. Then we selected five representative firms for each implied rating, AA, A, BBB, BB, B, and C. The implied rating is graded based on the five year CDS spread of the firm. Also, any two firms having the same rating are not included in the same industrial sector. For the CDS spreads we've used six different maturities, 1, 2, 3, 5, 7, and 10 years, since they are very liquidly traded credit derivatives among several credit derivatives.

The statistics of the selected data are summarized in Table 5.10 and 5.11. The spreads are represented as a percentage. Noticeably, data of most rating groups have very high standard deviations and maxima. This is a typical feature of the positively skewed data set and this coincides the highly positive skewness values as shown in the tables. This positive skewnesses seem to be originated from the high CDS spreads of the global financial crisis period.

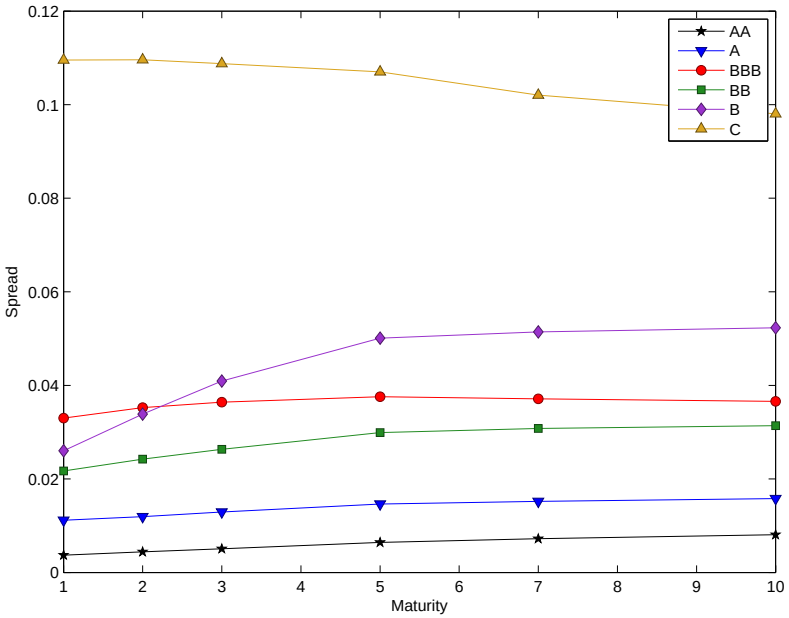
Figure 5.5 shows the term structures of the mean and median spreads for each rating group. In the graph of mean spreads, we can see that the spreads decrease as the implied rating improves when the maturity is fixed except for the group of BBB rating. The reason for this exception is that the mean value is sensitive to some large values. Especially, Ford Motor Company in the group of

Table 5.10 Basic statistics of the selected data set of the AA to BBB rating groups. The spreads are represented as a percentage (100 bp).

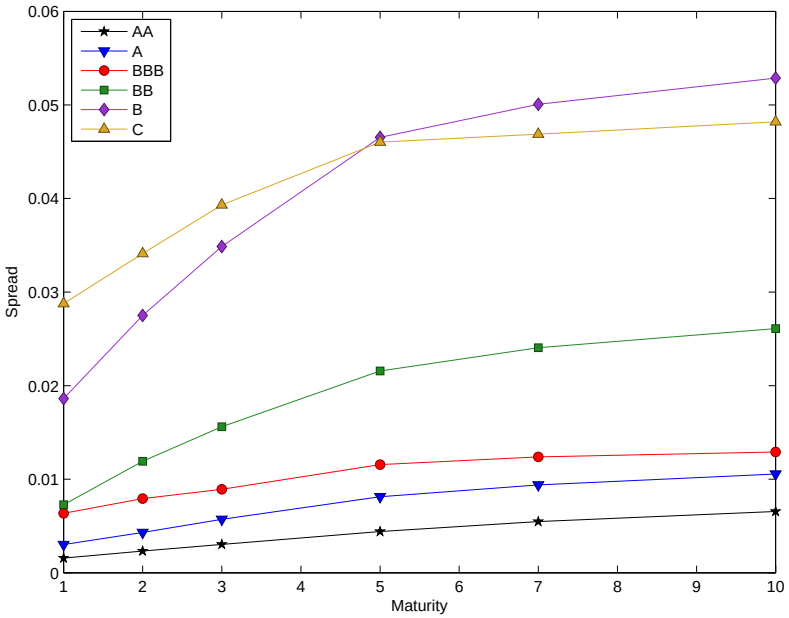
Rating	Statistics	Maturity					
		1Y	2Y	3Y	5Y	7Y	10Y
AA	mean	0.3705	0.4424	0.5077	0.6418	0.7211	0.8081
	std.dev.	0.6938	0.6909	0.6532	0.6446	0.6292	0.6173
	min	0.0140	0.0223	0.0305	0.0574	0.0825	0.1120
	median	0.1572	0.2316	0.3035	0.4410	0.5470	0.6544
	max	8.4001	7.8672	7.7486	7.3746	7.0579	6.8820
	skew	6.2316	5.7746	4.9098	4.1485	3.9504	3.7062
A	mean	1.1168	1.1935	1.2925	1.4633	1.5205	1.5799
	std.dev.	3.4282	2.9976	2.8149	2.4867	2.2196	2.0047
	min	0.0110	0.0200	0.0381	0.0677	0.0900	0.1023
	median	0.3021	0.4304	0.5712	0.8129	0.9392	1.0560
	max	67.7230	53.0722	45.6853	38.1119	33.2684	29.1860
	skew	8.9471	7.6979	6.9729	6.2172	5.7942	5.4023
BBB	mean	3.2994	3.5232	3.6397	3.7565	3.7107	3.6576
	std.dev.	12.0486	11.1026	10.2291	9.2389	8.5481	7.9098
	min	0.0447	0.0752	0.1000	0.1800	0.2117	0.2516
	median	0.6347	0.7921	0.8928	1.1566	1.2384	1.2909
	max	134.238	129.014	123.176	117.764	114.133	108.237
	skew	7.4621	7.1876	7.0362	7.0393	7.1650	7.2775

Table 5.11 Basic statistics of the selected data set of the BB to C rating groups. The spreads are represented as a percentage (100 bp).

Rating	Statistics	Maturity					
		1Y	2Y	3Y	5Y	7Y	10Y
BB	mean	2.1716	2.4243	2.6317	2.9900	3.0781	3.1377
	std.dev.	3.8042	3.4495	3.1998	2.9326	2.7774	2.6447
	min	0.0369	0.0658	0.0951	0.1730	0.2100	0.2773
	median	0.7279	1.1917	1.5603	2.1566	2.4054	2.6094
	max	58.6189	37.7039	34.7189	29.4581	26.3604	22.7197
	skew	4.4169	3.3698	2.8427	2.1328	1.8674	1.6544
B	mean	2.6014	3.3860	4.0913	5.0095	5.1440	5.2297
	std.dev.	2.2829	2.6063	2.8358	2.9178	2.7293	2.5829
	min	0.0950	0.3750	0.4500	0.7519	0.7537	0.8345
	median	1.8617	2.7500	3.4869	4.6540	5.0072	5.2866
	max	16.0751	18.0306	17.9076	18.3501	16.7750	15.2661
	skew	1.5652	1.5299	1.3985	1.0524	0.8066	0.6073
C	mean	10.95254	10.9582	10.8783	10.7015	10.2037	9.8036
	std.dev.	33.4680	29.4195	25.8971	22.8022	20.5104	18.7787
	min	0.0489	0.0857	0.1906	0.3719	0.5518	0.7164
	median	2.8764	3.4114	3.9314	4.6025	4.6884	4.8191
	max	393.177	429.800	402.680	397.940	267.801	239.643
	skew	7.3444	7.3754	6.9731	6.9465	6.6013	6.5255



(a)



(b)

Figure 5.5 The term structure of mean and median spreads for each rating group. (a) mean spreads (b) median spreads

BBB has very large maximum spreads values because the automotive industry was one of the industries that had a very severe situation in the financial crisis period. The graph of median spreads shows more typical structures than that of mean spreads. The spreads decrease as the implied rating improves except for the B and C grades when the maturity is greater than or equal to five years. In addition, the spreads increase when the maturity increases in the graph of median spreads while there are decreasing term structure graphs for some implied rating groups in the graph of mean spreads.

Experimental procedures

We predicted the CDS spreads for six different maturities using the CDS spreads of past 14 days and those past values are exploited as the input variables without any manipulations. Since there are six values of CDS spreads, one for each maturity, the total dimension of input variables is 84 and the total dimension of target variables is 6. This structure of data object is displayed in Figure 5.6. Then we divided the constructed data set into non-overlapping one-month subperiods. Since the period of the whole da ta set includes the contracts from January 2001 to February 2014, 158 subperiods were constructed in total for each firm.

For the prediction using nonparametric models, we used the roll-over strategy as follows. First, train the model with the first subperiod and test the model with the next subperiod. Repeating this process until the last subperiod is used as the test set. This roll-over strategy is briefly summarized in Figure 5.7. This roll-over strategy was also used to calibrate the parameters for the benchmark

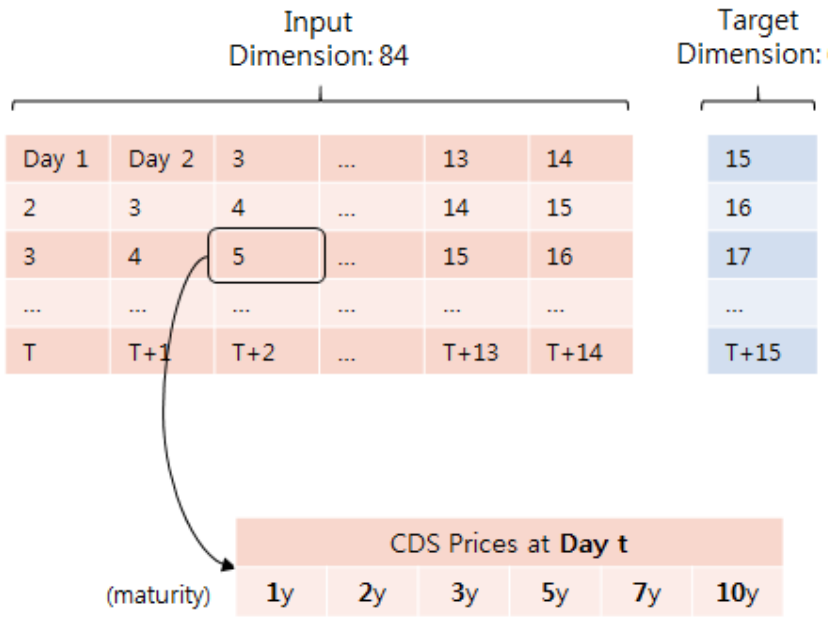


Figure 5.6 The structure of data set. Each instance has 84 input variables and 6 target variables.

model. The parameters of the benchmark model were selected as optimal to the train subperiod and applied to the test subperiod to measure the prediction performance.

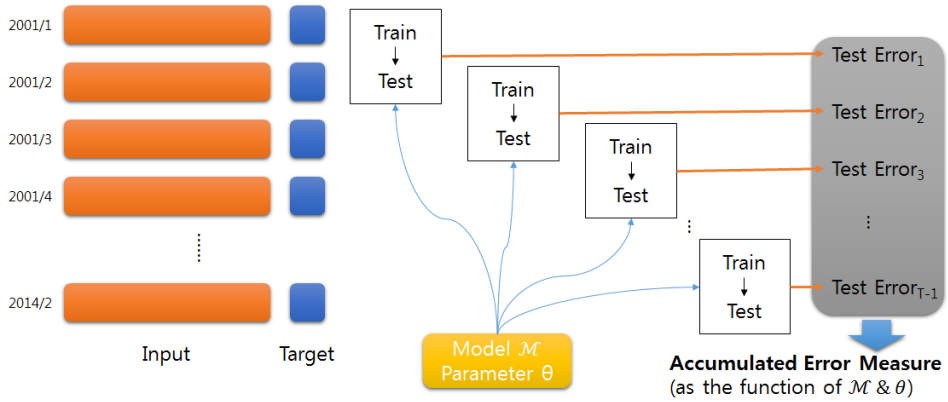


Figure 5.7 Roll-over prediction strategy of CDS spreads.

The parameters and other settings of nonparametric models were basically the same with that of the parametric benchmark model explained above. For ANNs and BNNs, the sigmoid function was used as an activation function except for the last layer for which the linear function is used as an activation function. For SVR and RVM, the radial basis kernel, $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x} - \mathbf{y}\|^2)$, was used for the basis kernel function. GP used constant mean and covariance functions with the Gaussian likelihood function for hyperparameters. The specific values of parameters and other variables were selected from the train subperiod, with an appropriate validation task, and applied to the test subperiod.

In addition, we applied three othre models: linear regression, ridge regression, and Cox-Ingersoll-Ross (CIR) model. For those models, the parameters

were selected appropriately to one subperiod and the errors were measured for the next subperiod with those parameters.

For measuring the performance, we took the average value of relative root-mean-squared error (RMSE) of all test sets. The relative RMSE for each test set is computed as

$$\text{Rel. RMSE} = \left[\frac{1}{|D|} \sum_{t \in D} \left\{ \frac{1}{|M|} \sum_{m \in M} \left(\frac{s(m,t) - \hat{s}(m,t)}{s(m,t)} \right)^2 \right\} \right]^{\frac{1}{2}} \tag{5.31}$$

where D is the set of dates that the data set includes, M is the set of maturities, $s(m,t)$ is the actual spread value for the maturity m and date t, and $\hat{s}(m,t)$ is the predicted value. The weights for the different maturities and dates are set to be all the same.

5.4.5 Experimental Results

For more analysis, we averaged the prediction results for each implied rating group and this results are shown in Table 5.13 and Figure 5.8. For the averaged results, BNNs showed the best performances for every group except AA-rated one where GP performed the best. SVR did not show a significantly different performance with the benchmark model (i.e. the constant intensity model) and this result coincides with the previous research of (Gündüz & Uhrig-Homburg, 2011) that only considered one value CDS spread with 5 year maturity. Especially, SVR was worse than the constant intensity model in the relatively low grade groups, BB and C, according to Figure 5.8. RVM performed well for high grade firms but did not show good results for low grade firms. It was also worse than the benchmark for C grade firms. In contrast, the other three nonparamet-

Table 5.12 Averaged relative RMSE of predicted CDS spreads for each firm. The **boldface** represents the best result and the diverged results due to bad calibration are remained as blanks.

Implied rating	Ticker	ANN	BNN	SVR	GP	RVM	LR	Ridge	CIR	Benchmark
	AMGN	0.1705	0.1611	0.4607	0.1562	**0.1536**	0.5963	0.7167	0.3046	0.4789
	CSX	0.1461	**0.1365**	0.2331	0.1382	**0.1315**	1.1684	0.2074		0.4969
AA	IBM	0.1410	0.1326	0.2760	0.1266	**0.1240**	0.7881	0.3177	0.2947	0.4846
	JPM	0.1745	0.1603	0.3051	0.1538	0.1578	0.3587	**0.1472**	0.3223	0.4677
	T	0.1424	**0.1361**	0.3021	0.1380	0.1733	0.4393	0.1405	0.2867	0.4644
	CAT	0.1466	0.1441	0.2166	**0.1396**	0.1425	0.5693	0.5106	0.2983	0.5225
	IP	0.1298	**0.1155**	0.2720	0.1159	0.1570	0.6043	0.7180	0.2850	0.4936
A	AIG	**0.1623**	0.1826	0.4528	0.1826	0.3062	0.5161	0.8462	0.3290	0.4653
	CU	0.1500	**0.1425**	0.3635	0.1500	0.2750	0.3789	0.5581		0.2175
	CNP	0.1194	0.1096	0.2329	0.1095	**0.1076**	1.6285	0.6914		0.4785
	CA	0.1363	**0.1281**	0.7150	0.1351	0.2749	0.5712	0.7401	0.3241	0.3528
	F	**0.1405**	0.1466	0.3884	0.1522	0.4380	2.5302	0.8142	0.3146	0.3591
BBB	MAY	0.1145	**0.1072**	0.2870	0.1151	0.1140	0.5402	0.3928		0.3837
	PH	**0.0878**	0.0897	0.2335	0.1123	**0.0862**	0.3568	0.2637		0.4962
	N	0.1322	0.1275	0.2875	0.1269	0.1244	0.4453	**0.1063**		0.4688
	KBH	**0.1319**	0.1320	0.2986	0.1380	0.2821	0.4242	0.1530	0.3140	0.2371
	HRB	0.1763	0.1954	0.8024	0.2030	0.2192	0.4256	**0.1413**		0.2989
BB	SLMA	0.1581	**0.1556**	0.4005	0.1611	0.4044	1.2397	0.5504	0.2995	0.4466
	DPL	0.1223	0.1134	0.2765	0.1110	**0.1089**	0.6359	0.4322	0.2578	0.4480
	EP	0.1529	**0.1420**	0.7713	0.1451	0.3373	0.5246	0.6811	0.3186	0.4221
	AMKR	0.1295	**0.1279**	0.2666	0.1323	0.2453	0.3968	0.8413		0.4788
	INTEL	0.1270	**0.1160**	0.1671	0.1184	0.2558	1.2045	0.5986		0.4831
B	PLCOAL	**0.0305**	0.0357	0.1652	0.0355	0.0353	0.4114	0.1745	0.2687	0.4690
	THC	0.1123	**0.1030**	0.2761	0.1048	0.2402	0.4153	0.2279		0.4169
	FST	**0.0975**	0.1019	0.1910	0.1043	0.1086	0.5108	0.7214	0.2882	0.5272
	BOW	0.1219	**0.1105**	0.4008	0.1123	0.3745	0.8040	0.6365	0.2980	0.3770
	DYN	**0.2469**	0.2353	0.8476	0.2386	0.5580	0.5942	**0.2003**		0.3634
C	AMR	**0.1032**	0.1124	0.7381	0.1132	0.5850	0.5172	0.7695		0.2922
	VC	0.1687	**0.1553**	0.4131	0.1774	0.5004	0.7558	0.3767	0.3333	0.3185
	JCP	0.1733	**0.1518**	0.2305	0.1540	0.2838	0.5363	0.1969	0.3188	0.4748

Table 5.13 Averaged relative RMSE of predicted CDS spreads for each implied rating group. The **boldface** represents the best result.

Implied rating	ANN	BNN	SVR	GP	RVM	LR	Ridge	CIR	Benchmark
AA	0.1549	0.1453	0.3154	**0.1426**	0.1480	0.6701	0.3059	0.3020	0.4785
A	0.1416	**0.1388**	0.3076	0.1415	0.1977	0.7394	0.6648	0.3041	0.4355
BBB	0.1223	**0.1198**	0.2712	0.1283	0.2075	0.8887	0.4634	0.3193	0.4121
BB	0.1483	**0.1477**	0.5099	0.1517	0.2704	0.6500	0.3916	0.2975	0.3705
B	0.0994	**0.0969**	0.2132	0.0991	0.1770	0.5878	0.5127	0.2784	0.4750
C	0.1628	**0.1531**	0.5260	0.1591	0.4604	0.6415	0.4360	0.3167	0.3652
Average	0.1382	**0.1336**	0.3757	0.1370	0.2435	0.6963	0.4624	0.3030	0.4228

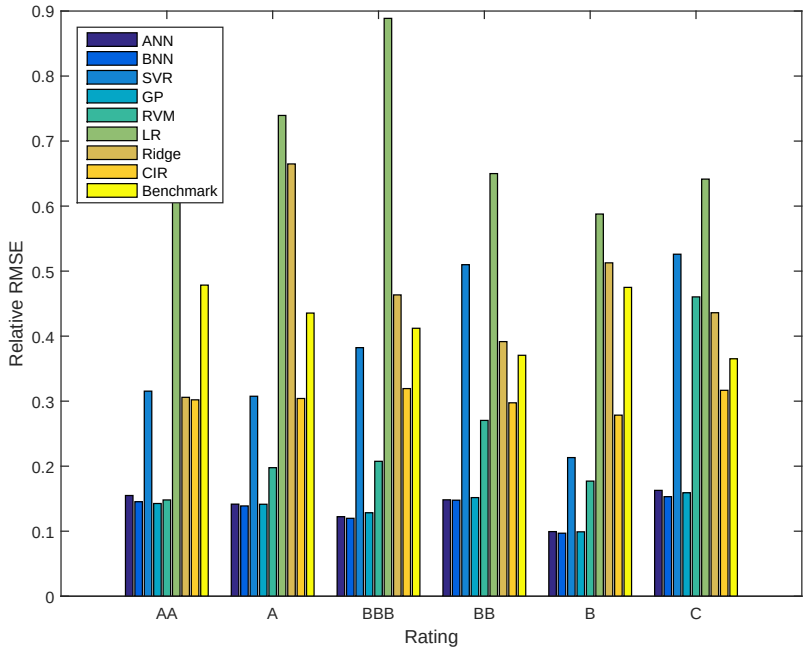
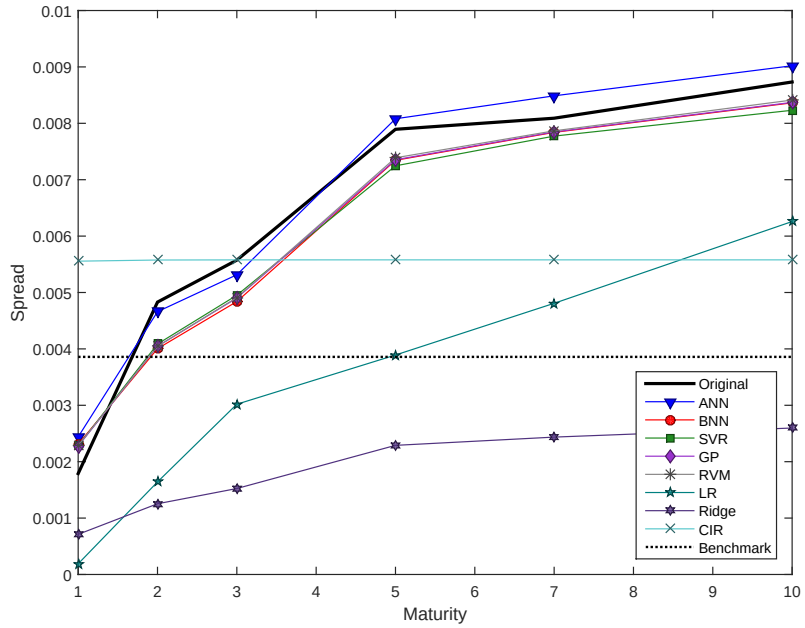


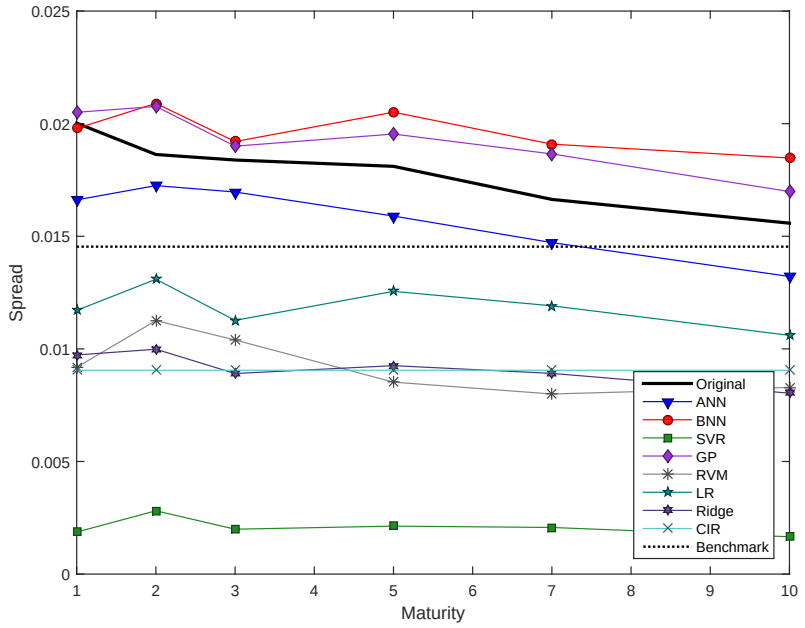
Figure 5.8 Averaged relative RMSE of predicted CDS spreads for each implied rating group.

ric models showed much better prediction accuracy than the benchmark model with not much difference among them for all the implied rating groups.

Figure 5.9 shows two examples of the CDS spread prediction, one from the high grade firms and the other from the low grade firms. The first example is selected as the spreads of AT&T Incorporation which belongs to the group of AA rated firms. We can observe that the all nonparametric model predicted well for this example. However, in the second example, the predicted spreads of H&R Block Incorporation which belongs to the group of BB rated firms, it is obviously presented that SVR performed worse than the other models.



(a)



(b)

Figure 5.9 Examples of the term structure of predicted spreads. Original refers to the actual spread from the market. (a) T from AA rating group (b) HRB from BB rating group.

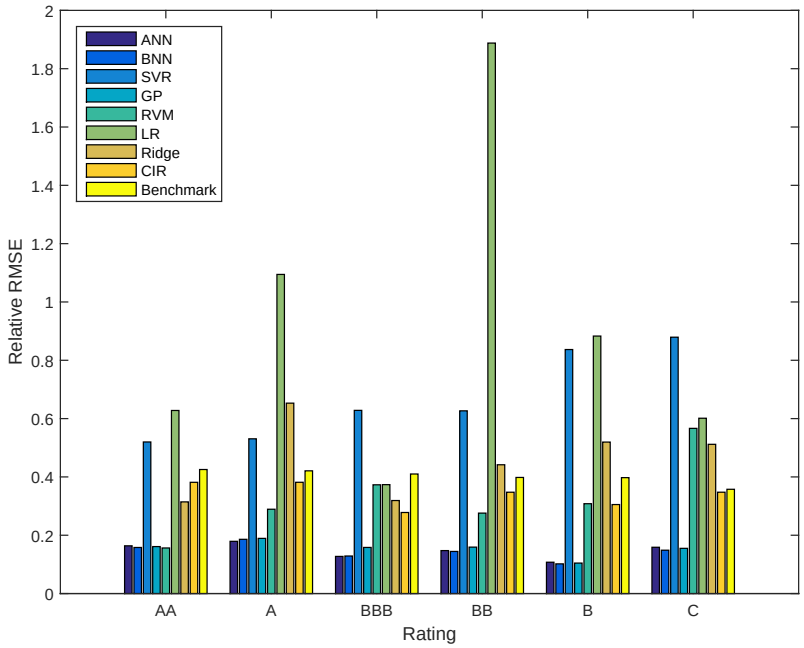


Figure 5.10 Averaged relative RMSE of predicted CDS spreads for the global financial crisis period.

Additionally, to compare the predictive performance of the models in financial crisis, we applied the prediction models to the global financial crisis period from 2007 to 2009. The results are shown in Figure 5.10. The nonparametric models except the SVR consistently showed significantly better performances than the benchmark model. However, SVR performed worse than the benchmark even for the high grades unlike the moderate preiod. In the crisis period, CDS spreads of most firms increased sharply thus even the high grade firms might have large spread values than the low grade firms of the moderate period. Thus, SVR which showed bad performances for low grade firms in the

whole period did not forecast well even for the high grade firms in the crisis period. Therefore, SVR might not be useful in predicting the credit risks of low-rated firms or in making forecasts during a crisis when the credit risks of most firms are large.

5.5 Chapter Summary

In this chapter, we conducted an empirical study on the predictive performance of nonparametric machine learning models for market impact costs and CDS spread prediction. This study has several features. First, five state-of-arts nonparametric machine learning models including ANNs, BNNs, SVR, GP regression, and RVM have been compared to verify their performances to predict market impacts and CDS spreads with a benchmark parametric model whereas the previous studies were usually focused on parametric models or additional one nonparametric model. Second, the data set used in this study is very extensive. For the market impact prediction, the total number of transactions exceeds 18 million. 17 firms were selected from each indices of large, mid, and small cap firms, while the previous studies mostly focused on the large cap firms. The market impact prediction in this chapter used independent variables from single transactions thus it has advantages to be applied to technological high-frequency trades compared to the previous studies analyzing large size trades. The active learning model developed in chapter 4 was also applied to this task. For the CDS spread prediction, the data set contains daily contract data from January 2001 to February 2014 including the global financial crisis

period. Also, the data sets for two applications in this chapter contain various kinds of firms. CDS spreads of various rating of firms, from AA to C, were used for the prediction whereas most of the earlier studies focused on AA or A ratings. Finally, the prediction for the CDS spreads with different maturities were conducted simultaneously, i.e. the multi-valued CDS spreads, not just the spread of the one maturity since it is very important to analyze the spreads of different maturities at a specific time for any firms for market users.

As a results of this study, most nonparametric machine learning models possibly except the SVR and the RVM outperformed robustly than the benchmark parametric models for both tasks. Especially, BNNs showed the best performances in usual, whereas SVR and RVMs sometimes performed worse than the benchmark model.

There are still remaining possible tasks that can be improved by the nonparametric learning approaches. For example, the path-dependent multi-asset derivative pricing is one of difficult task for the existing parametric models. The learning models can help to solve this problem. As mentioned, one of the advantage of learning models is they can take any variables as input values. Thus, including derived variables from the original variables or non-market variables like news or text data can also improve the results of the tasks conducted in this chapter.

Chapter 6

Conclusion

6.1 Summary of research

As an era of big data arises, sparse learning models are explored again since they requires both small storage space and computational time with the comparable performances. In this dissertation, we developed two novel sparse learning models using ARD prior distribution and applied both sparse and non sparse learning models to financial problems to improve the estimation and prediction performances.

In chapter 3, we propose a sparse support-based clustering method whose support function is represented by a small number of kernel basis vectors. The proposed method applied the ARD prior and the variable GP noise to the GP regression model to build a sparse support function. The method assigns the hypothetical output values, which are not directly related to the cluster labels, of the clustering data sets to obtain a tractable likelihood function in the GP regression model to determine hyper-functions and hyper-parameters efficiently. The rigorous theoretical background for constructed support function can indeed estimate the support of the given data distribution is also given in addition.

The proposed method has several features compared with the findings of previous studies on support-based clustering methods. First, the constructed support function is represented by a significantly smaller number of kernel center points than the other methods. Second, the kernel center points are automatically selected from the given data points during the training process and can represent the rest of the data, playing a similar role as representative points or exemplars. Third, the simple nearest neighbor method can naturally be used as the labeling method to boost the labeling in the training phase as well as the clustering in the test phase. Finally, the operability of clustering and characteristics explained previously were verified through several experiments including some benchmark and real clustering data sets, image segmentation, and handwritten digits by determining its representative data.

Then, we develop the novel active learning method for transductive sparse Bayesian regression in chapter 4. We first propose a transductive and generalized version of RVM in which the relevance vectors of the constructed model are selected from the unlabeled data points as well as the labeled data points. Next, we propose three querying strategies for the active selection of the labeled point set using only the relevance vectors automatically obtained from the developed model, thereby making an additional process for active learning unnecessary. The proposed active learning algorithm is completed by repeating the two previously-mentioned procedures until the model converges or one of the stopping criteria is satisfied. The proposed method outperformed the random selection algorithm for both artificial and real data sets, whereas it did not perform well compared with the full RVM model that used the whole data

points as labeled. The three strategies showed different characteristics when applied to the real data sets. The first querying strategy, *querying all unlabeled relevance vectors*, and the last querying strategy, *querying the farthest relevance vector*, showed significantly small MSEs and standard errors but they required a large number of labeled points to converge. The second strategy, *querying the most uncertain relevance vector*, converged with a small number of labeled points, which means that the labeling cost can be minimized, while MSEs and standard errors from this strategy were higher than those of the other strategies.

Finally, in chapter 5, we applied the state-of-the-art learning models including sparse kernel machines to two financial problems: predicting the market impact costs and the credit default swap spreads. A market impact cost, one of the implicit transaction cost, is a transaction cost caused by the price difference between one before the transaction and one of actual execution. Since there had not been we exploited learning models to predict the market impact costs, we tried to estimate and predict the market impacts of equity market. In addition to the learning models, we chose and applied one parametric model to single transaction data of US stock markets. In this study, we used single transaction data from the firms belonging to large, mid, or small cap index and analyzed them both separately and altogether, while the previous studies usually focused on the trade of large cap firms. Credit default swap is the most liquidly traded credit derivative, which is an insurance contract between one party holding a risk asset and its counterparty ensuring the compensation when the default of the risk asset occurs. Also for these credit default swaps, there had not been a research result of applying several machine learning methods and comparing

the result among one another. We applied several machine learning methods and one parametric benchmark model to predict spreads of daily CDS contracts from 2001 to 2014. For both tasks, the machine learning models showed better performances than the parametric benchmark models. In addition, for the market impact prediction, we applied the active learning algorithm developed in chapter 4 gained the results that coincide the results for other data sets in chapter 4.

6.2 Future Work

For each topic included in this dissertation, there are several directions for further research to improve the proposed methods and their results, and investigate the related research topic.

Possible future work to imporve the proposed sparse support clustering method proposed in chapter 3 are as follows. First, in case that the precision of clustering is important, more sophisticated labeling algorithms, such as complete graph approach(Ben-Hur et al., 2002) and dynamic system approach(J. Lee & Lee, 2005, 2006), can be used for the proposed support function rather than the naïve nearest neighbor approach which concentrates on reducing labeling time. Next, although the selected parameters σ and L result in variation in the shape and number of clusters, determining appropriate parameters is sometime difficult. If a method or criterion for selecting the parameters is recommended for the proposed methods as for previous methods(K.-P. Wu & Wang, 2009), it may be helpful to use the proposed method. Finally, the number and dimen-

sion of the data set can increase in the application to some real data sets. If the algorithm of the proposed method is parallelized, then the proposed method can be effectively applied to large data sets with powerful parallel computing methods that have been developed recently.

There exist also possible ways to improve the active learning algorithm in chapter 4. First, the user should select the initial set of labeled points. In this paper, we randomly selected three points from given data points and the method worked successfully. However, other sophisticated sampling methods, including clustering approaches, may improve the performances of the first few iterations. The stopping criterion is another issue that the user must decide on. It is difficult to predict when the model converges, however, we observed that the proposed methods performed significantly better than the random selection after several iterations, and the MSEs at a few iterations before convergence were comparable to that at convergence. Thus, it seems that an earlier stopping than the optimal model may have prevented the occurrence of severe problems. Finally, other querying strategies which use the information criteria can be developed. Similar to the suggestions of previous studies(Zhang & Oles, 2000; Schein & Ungar, 2007; Settles & Craven, 2008), a querying strategy based on information theory can be applied to the transductive GRVM developed in this study in order to construct another active learning algorithm for sparse Bayesian regression.

As mentioned in chapter 5, one of the advantage of learning models is they can take any variables as input values. Including derived variables from the original variables or non-market variables like news or text data can also im-

prove the results of the tasks conducted in this chapter. There are also still remaining unsolved financial tasks that can be improved possibly by the non-parametric learning approaches. For example, the path-dependent multi-asset derivative pricing is one of difficult task for the existing parametric models. The combination of learning models and traditional solutions for the problem like Monte Carlo method can help to solve this problem.

Bibliography

Abe, H., & Mamitsuka, N. (1998). Query learning strategies using boosting and bagging. In *Machine learning: Proceedings of the fifteenth international conference (icml'98)* (p. 1).

Almgren, R., Thum, C., Hauptmann, E., & Li, H. (2005). Direct estimation of equity market impact. *Risk*, *18*(7), 58–62.

Arbelaez, P., Fowlkes, C., & Martin, D. (2007). The berkeley segmentation dataset and benchmark. *see http://www. eecs. berkeley. edu/Research/Projects/CS/vision/bsds*.

Asuncion, A., & Newman, D. (2007). *Uci machine learning repository.*

Bakshi, G., Madan, D., & Zhang, F. X. (2006). Investigating the role of systematic and firm-specific factors in default risk: Lessons from empirically evaluating credit risk models. *The Journal of Business*, *79*(4), 1955–1987.

Ban, T., & Abe, S. (2004). Spatially chunking support vector clustering algorithm. In *Neural networks, 2004. proceedings. 2004 ieee international joint conference on* (Vol. 1).

Ben-Hur, A., Horn, D., Siegelmann, H. T., & Vapnik, V. (2002). Support vector

clustering. *The Journal of Machine Learning Research*, *2*, 125–137.

Bershova, N., & Rakhlin, D. (2013). The non-linear market impact of large trades: Evidence from buy-side order flow. *Quantitative Finance*, *13*(11), 1759–1778.

Bhattacharyya, S., Pictet, O. V., & Zumbach, G. (2002). Knowledge-intensive genetic discovery in foreign exchange markets. *Evolutionary Computation, IEEE Transactions on*, *6*(2), 169–181.

Bikker, J. A., Spierdijk, L., Hoevenaars, R. P., & Van der Sluis, P. J. (2008). Forecasting market impact costs and identifying expensive trades. *Journal of Forecasting*, *27*(1), 21–39.

Bikker, J. A., Spierdijk, L., & Van Der Sluis, P. J. (2007). Market impact costs of institutional equity trades. *Journal of International Money and Finance*, *26*(6), 974–1000.

Bishop, C. M. (2006). *Pattern recognition and machine learning* (Vol. 4) (No. 4). springer New York.

Black, F., & Cox, J. C. (1976). Valuing corporate securities: Some effects of bond indenture provisions. *The Journal of Finance*, *31*(2), 351–367.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on computational learning theory* (pp. 144–152).

Bottolo, L., Richardson, S., et al. (2010). Evolutionary stochastic search for bayesian model exploration. *Bayesian Analysis*, *5*(3), 583–618.

Bouchaud, J.-P., Gefen, Y., Potters, M., & Wyart, M. (2004). Fluctuations and response in financial markets: the subtle nature of 'random'price changes. *Quantitative Finance*, *4*(2), 176–190.

Bühler, W., & Trapp, M. (2009). *Time-varying credit risk and liquidity premia in bond and cds markets* (Tech. Rep.). CFR working paper.

Burbidge, R., Rowland, J. J., & King, R. D. (2007). Active learning for regression based on query by committee. In *Intelligent data engineering and automated learning-ideal 2007* (pp. 209–218). Springer.

Cao, L.-J., & Tay, F. E. H. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *Neural Networks, IEEE Transactions on*, *14*(6), 1506–1518.

Ceperic, V., Gielen, G., & Baric, A. (2012). Sparse multikernel support vector regression machines trained by active learning. *Expert Systems with Applications*, *39*(12), 11029–11035.

Ceperic, V., Gielen, G., & Baric, A. (2014). Sparse ε-tube support vector regression by active learning. *Soft Computing*, *18*(6), 1113–1126.

Chaloner, K., & Verdinelli, I. (1995). Bayesian experimental design: A review. *Statistical Science*, 273–304.

Chen, S., & Wigger, J. (1995). Fast orthogonal least squares algorithm for efficient subset model selection. *IEEE Transactions on Signal Procesing*, *43*(7), 1713–1715.

Chen, W.-H., Shih, J.-Y., & Wu, S. (2006). Comparison of support-vector machines and back propagation neural networks in forecasting the six major asian stock markets. *International Journal of Electronic Finance*, *1*(1), 49–67.

Cohn, D. A. (1996). Neural network exploration using optimal experiment design. *Neural networks*, *9*(6), 1071–1083.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of artificial intelligence research*.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Cox, D. R. (1955). Some statistical methods connected with series of events. *Journal of the Royal Statistical Society. Series B (Methodological)*, 129–164.

Cox, J. C., Ingersoll Jr, J. E., & Ross, S. A. (1985). A theory of the term structure of interest rates. *Econometrica: Journal of the Econometric Society*, 385–407.

Cressie, N. A. (1993). *Statistics for spatial data* (Vol. 900). Wiley New York.

Csató, L., & Opper, M. (2002). Sparse on-line gaussian processes. *Neural computation*, *14*(3), 641–668.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, *2*(4), 303–314.

Demir, B., & Bruzzone, L. (2014). A multiple criteria active learning method for support vector regression. *Pattern Recognition*, *47*(7), 2558–2567.

Douak, F., Melgani, F., & Benoudjit, N. (2013). Kernel ridge regression with active learning for wind speed prediction. *Applied Energy*, *103*, 328–340.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. *Advances in neural information processing systems*, *9*, 155–161.

Duffee, G. R. (1999). Estimating the price of default risk. *Review of Financial Studies*, *12*(1), 197–226.

Duffie, D., & Singleton, K. J. (2012). *Credit risk: pricing, measurement, and management*. Princeton University Press.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*, *32*(2), 407–499.

Eom, Y. H., Helwege, J., & Huang, J.-z. (2004). Structural models of corporate bond pricing: An empirical analysis. *Review of Financial studies*, *17*(2), 499–544.

Figueiredo, M. A. (2003). Adaptive sparseness for supervised learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *25*(9), 1150–1159.

Finger, C., Finkelstein, V., Lardy, J.-P., Pan, G., Ta, T., & Tierney, J. (2002). Creditgrades technical document. *RiskMetrics Group*, 1–51.

Flaherty, P., Arkin, A., & Jordan, M. I. (2005). Robust design of biological experiments. In *Advances in neural information processing systems* (pp. 363–370).

Foresee, F. D., & Hagan, M. T. (1997). Gauss-newton approximation to bayesian learning. In *Proceedings of the 1997 international joint conference on neural networks* (Vol. 3, pp. 1930–1935).

Frino, A., Bjursell, J., Wang, G. H., & Lepone, A. (2008). Large trades and intraday futures price behavior. *Journal of Futures Markets*, *28*(12), 1147–1181.

Fu, W. J. (1998). Penalized regressions: the bridge versus the lasso. *Journal of computational and graphical statistics*, *7*(3), 397–416.

Fu, Y., Zhu, X., & Li, B. (2013). A survey on instance selection for active learning. *Knowledge and information systems*, *35*(2), 249–283.

Gabaix, X., Gopikrishnan, P., Plerou, V., & Stanley, H. E. (2003). A theory of power-law distributions in financial market fluctuations. *Nature*, *423*(6937), 267–270.

Girolami, M. (2002). Mercer kernel-based clustering in feature space. *Neural Networks, IEEE Transactions on*, *13*(3), 780–784.

Gündüz, Y., & Uhrig-Homburg, M. (2011). Predicting credit default swap prices with financial and pure data-driven approaches. *Quantitative Finance*, *11*(12), 1709–1727.

Guo, J., Chen, H., Sun, Z., & Lin, Y. (2004). A novel method for protein secondary structure prediction using dual-layer svm and profiles. *PROTEINS: Structure, Function, and Bioinformatics*, *54*(4), 738–743.

Guo, Y., & Greiner, R. (2007). Optimistic active-learning using mutual information. In *Ijcai* (Vol. 7, pp. 823–829).

Han, G.-S., & Lee, J. (2008). Prediction of pricing and hedging errors for equity linked warrants with gaussian process models. *Expert Systems with Applications*, *35*(1), 515–523.

Hansen, M. S., Sjöstrand, K., Ólafsdóttir, H., Larsson, H. B., Stegmann, M. B., & Larsen, R. (2007). Robust pseudo-hierarchical support vector clustering. In *Image analysis* (pp. 808–817). Springer.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., & Tibshirani, R. (2009). *The elements of statistical learning* (Vol. 2) (No. 1). Springer.

Hoi, S. C., Jin, R., & Lyu, M. R. (2009). Batch mode active learning with applications to text categorization and image retrieval. *Knowledge and Data Engineering, IEEE Transactions on*, *21*(9), 1233–1248.

Huang, J. C., Morris, Q. D., & Frey, B. J. (2007). Bayesian inference of microrna targets from sequence and expression data. *Journal of Computational Biology*, *14*(5), 550–563.

Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision support systems*, *37*(4), 543–558.

Huberman, G., & Stanzl, W. (2005). Optimal liquidity trading. *Review of Finance*, *9*(2), 165–200.

Hull, J. J. (1994). A database for handwritten text recognition research. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *16*(5), 550–554.

Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, *49*(3), 851–889.

Jarrow, R. A., & Turnbull, S. M. (1995). Pricing derivatives on financial securities subject to credit risk. *The journal of finance*, *50*(1), 53–85.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Icml* (Vol. 99, pp. 200–209).

Jones, E. P., Mason, S. P., & Rosenfeld, E. (1984). Contingent claims analysis of corporate capital structures: An empirical investigation. *The Journal of Finance*, *39*(3), 611–625.

Jung, K.-H., Lee, D., & Lee, J. (2010). Fast support-based clustering method for large-scale problems. *Pattern Recognition*, *43*(5), 1975–1983.

Kapoor, A., Grauman, K., Urtasun, R., & Darrell, T. (2007). Active learning with gaussian processes for object categorization. In *Computer vision, 2007. iccv 2007. ieee 11th international conference on* (pp. 1–8).

Kato, T. (2014). An optimal execution problem with market impact. *Finance and Stochastics*, *18*(3), 695–732.

Kim, H.-C., & Lee, J. (2007). Clustering based on gaussian processes. *Neural computation*, *19*(11), 3088–3107.

Kim, K., Son, Y., & Lee, J. (2015). Voronoi cell-based clustering using a kernel support. *Knowledge and Data Engineering, IEEE Transactions on*, *27*(4), 1146–1156.

Kim, K.-j., & Ahn, H. (2012). A corporate credit rating model using multi-class support vector machines with an ordinal pairwise partitioning approach. *Computers & Operations Research*, *39*(8), 1800–1811.

Kim, S. H., & Noh, H. J. (1997). Predictability of interest rates using data mining tools: a comparative analysis of korea and the us. *Expert Systems with Applications*, *13*(2), 85–95.

King, R. D., Rowland, J., Oliver, S. G., Young, M., Aubrey, W., Byrne, E., ... Soldatova, L. N. (2009). The automation of science. *Science*, *324*(5923), 85–89.

King, R. D., Whelan, K. E., Jones, F. M., Reiser, P. G., Bryant, C. H., Muggleton, S. H., ... Oliver, S. G. (2004). Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, *427*(6971), 247–252.

Kissell, R. (2013). *The science of algorithmic trading and portfolio management*. Academic Press.

Kissell, R., Glantz, M., & Malamut, R. (2003). *Optimal trading strategies: quantitative approaches for managing market impact and trading risk*. Amacom.

Krause, A., & Guestrin, C. (2007). Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In *Proceedings of the 24th international conference on machine learning* (pp. 449–456).

Krishnapuram, B., Carin, L., Figueiredo, M. A., & Hartemink, A. J. (2005). Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *27*(6), 957–968.

Lang, K. (1995). Newsweeder: Learning to filter netnews. In *Proceedings of the 12th international conference on machine learning* (pp. 331–339).

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, D., & Lee, J. (2007). Equilibrium-based support vector machine for semisupervised classification. *Neural Networks, IEEE Transactions on*, *18*(2), 578–583.

Lee, D., & Lee, J. (2010). Dynamic dissimilarity measure for support-based clustering. *Knowledge and Data Engineering, IEEE Transactions on*, *22*(6), 900–905.

Lee, J., & Lee, D. (2005). An improved cluster labeling method for support vector clustering. *IEEE Transactions on pattern analysis and machine intelligence*, *27*(3), 461–464.

Lee, J., & Lee, D. (2006). Dynamic characterization of cluster structures for robust and inductive support vector clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *28*(11), 1869–1874.

Lee, S.-H., & Daniels, K. M. (2012). Gaussian kernel width exploration and cone cluster labeling for support vector clustering. *Pattern Analysis and Applications*, *15*(3), 327–344.

Lee, Y.-C. (2007). Application of support vector machines to corporate credit rating prediction. *Expert Systems with Applications*, *33*(1), 67–74.

Lewis, D. D., & Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the eleventh international conference on machine learning* (pp. 148–156).

Liao, S.-H., & Chou, S.-Y. (2013). Data mining investigation of co-movements

on the taiwan and china stock markets for future investment portfolio. *Expert Systems with Applications*, *40*(5), 1542–1554.

Lillo, F., Farmer, J. D., & Mantegna, R. N. (2003). Econophysics: Master curve for price-impact function. *Nature*, *421*(6919), 129–130.

Liu, Y. (2004). Active learning with support vector machine applied to gene expression data for cancer classification. *Journal of chemical information and computer sciences*, *44*(6), 1936–1941.

Lyden, S., & Saraniti, D. (2001). An empirical examination of the classical theory of corporate security valuation. *Available at SSRN 271719*.

MacKay, D. J. (1992). Bayesian interpolation. *Neural computation*, *4*(3), 415–447.

Matsumoto, M., & Hori, J. (2014). Classification of silent speech using support vector machine and relevance vector machine. *Applied Soft Computing*, *20*, 95–102.

McCallum, A. K., & Nigamy, K. (1998). Employing em and pool-based active learning for text classification. In *Machine learning: Proceedings of the fifteenth international conference, icml.*

Merton, R. C. (1974). On the pricing of corporate debt: The risk structure of interest rates. *The Journal of Finance*, *29*(2), 449–470.

Micchelli, C. A. (1984). *Interpolation of scattered data: distance matrices and*

conditionally positive definite functions. Springer.

Moskovitch, R., Nissim, N., Stopel, D., Feher, C., Englert, R., & Elovici, Y. (2007). Improving the detection of unknown computer worms activity using active learning. In *Ki 2007: Advances in artificial intelligence* (pp. 489–493). Springer.

Murphy, K., & Dunham, M. (2008). Pmtk: Probabilistic modeling toolkit. In *Neural information processing systems (nips) workshop on probabilistic programming*.

Naveen, N. (2012). Application of relevance vector machines in real time intrusion detection. *Editorial Preface*, *3*(9).

Nesterov, Y. (2004). *Introductory lectures on convex optimization* (Vol. 87). Springer Science & Business Media.

Ogden, J. P. (1987). Determinants of the ratings and yields on corporate bonds: Tests of the contingent claims model. *Journal of Financial Research*, *10*(4), 329–340.

O'Hara, R. B., Sillanpää, M. J., et al. (2009). A review of bayesian variable selection methods: what, how and which. *Bayesian analysis*, *4*(1), 85–117.

Paisley, J., Liao, X., & Carin, L. (2010). Active learning and basis selection for kernel-based linear models: A bayesian perspective. *Signal Processing, IEEE Transactions on*, *58*(5), 2686–2700.

Park, H., Kim, N., & Lee, J. (2014). Parametric models and non-parametric machine learning models for predicting option prices: Empirical comparison study over kospi 200 index options. *Expert Systems with Applications*, *41*(11), 5227–5237.

Park, H., & Lee, J. (2012). Forecasting nonnegative option price distributions using bayesian kernel methods. *Expert Systems with Applications*, *39*(18), 13243–13252.

Park, J., Ji, X., Zha, H., & Kasturi, R. (2004). Support vector clustering combined with spectral graph partitioning. In *Pattern recognition, 2004. icpr 2004. proceedings of the 17th international conference on* (Vol. 4, pp. 581–584).

Plerou, V., Gopikrishnan, P., Gabaix, X., & Stanley, H. (2002). Quantifying stock-price response to demand fluctuations. *Physical review. E, Statistical, nonlinear, and soft matter physics*, *66*(2 Pt 2), 027104.

Puma-Villanueva, W. J., Bezerra, G. B., Lima, C. A., & Zuben, F. (2005). Improving support vector clustering with ensembles. In *Proceedings of international joint conference on neural networks* (pp. 13–15).

Rasmussen, C. E. (1996). *Evaluation of gaussian processes and other methods for non-linear regression* (Unpublished doctoral dissertation). University of Toronto.

Rattray, M., Stegle, O., Sharp, K., & Winn, J. (2009). Inference algorithms and

learning theory for bayesian sparse factor analysis. In *Journal of physics: Conference series* (Vol. 197, p. 012002).

Ribeiro, B., Vieira, A., & das Neves, J. C. (2006). Sparse bayesian models: Bankruptcy-predictors of choice? In *Ijcnn* (pp. 3377–3381).

Rosenblatt, F. (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms* (Tech. Rep.). DTIC Document.

Roy, N., & McCallum, A. (2001). Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). *Learning internal representations by error propagation* (Tech. Rep.). DTIC Document.

Sabuncu, M. R., Konukoglu, E., Initiative, A. D. N., et al. (2014). Clinical prediction from structural brain mri scans: A large-scale empirical study. *Neuroinformatics*, 1–16.

Scheffer, T., Decomain, C., & Wrobel, S. (2001). Active hidden markov models for information extraction. In *Advances in intelligent data analysis* (pp. 309–318). Springer.

Schein, A. I., & Ungar, L. H. (2007). Active learning for logistic regression: an evaluation. *Machine Learning*, *68*(3), 235–265.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural*

computation, *13*(7), 1443–1471.

Schönbucher, P. J. (2003). *Credit derivatives pricing models: models, pricing and implementation*. John Wiley & Sons.

Seo, S., Wallat, M., Graepel, T., & Obermayer, K. (2000). Gaussian process regression: Active data selection and test point rejection. In *Mustererkennung 2000* (pp. 27–34). Springer.

Settles, B. (2010). Active learning literature survey. *University of Wisconsin, Madison*, *52*(55-66), 11.

Settles, B., & Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 1070–1079).

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on computational learning theory* (pp. 287–294).

Shuib, N. L. M., Ismail, M. H., Chiroma, H., Abdullah, R., Shuib, A. S. M., & Pahme, N. F. M. (2014). Data mining approach: Relevance vector machine for the classification of learning style based on learning objects. In *Proceedings of the 2014 uksim-amss 16th international conference on computer modelling and simulation* (pp. 170–175).

Silva, C., & Ribeiro, B. (2007). Combining active learning and relevance vector machines for text classification. In *Machine learning and applications,*

2007. icmla 2007. sixth international conference on (pp. 130–135).

Simon, P. (2013). *Too big to ignore: The business case for big data*. John Wiley & Sons.

Snelson, E., & Ghahramani, Z. (2006). Sparse gaussian processes using pseudo-inputs.

Son, Y., Noh, D.-j., & Lee, J. (2012). Forecasting trends of high-frequency kospi200 index data using learning classifiers. *Expert Systems with Applications*, *39*(14), 11607–11615.

Soussen, C., Idier, J., Brie, D., & Duan, J. (2011). From bernoulli–gaussian deconvolution to sparse signal restoration. *Signal Processing, IEEE Transactions on*, *59*(10), 4572–4584.

Spectral clustering website. (n.d.). `http://ai.stanford.edu/~btaskar/ocr/`. (Accessed: 2015-05-11)

Tax, D. M., & Duin, R. P. (1999). Support vector domain description. *Pattern recognition letters*, *20*(11), 1191–1199.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Ticknor, J. L. (2013). A bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications*, *40*(14), 5501–5506.

Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector

machine. *The journal of machine learning research*, *1*, 211–244.

Tipping, M. E., & Faul, A. C. (2003). Fast marginal likelihood maximisation for sparse bayesian models. In *Proceedings of the ninth international workshop on artificial intelligence and statistics* (Vol. 1).

Tong, S., & Chang, E. (2001). Support vector machine active learning for image retrieval. In *Proceedings of the ninth acm international conference on multimedia* (pp. 107–118).

Tong, S., & Koller, D. (2002). Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, *2*, 45–66.

Tsang, I. W., Kwok, J. T., & Cheung, P.-M. (2005). Core vector machines: Fast svm training on very large data sets. In *Journal of machine learning research* (pp. 363–392).

Ultsch, A. (2005). Clustering with som: U*c. In *Proceedings of workshop on self-organizing maps (2005), pp. 75-82 key: citeulike:1304144* (pp. 75–82).

Vapnik, V. (2000). *The nature of statistical learning theory*. Springer Science & Business Media.

Vasicek, O. (1977). An equilibrium characterization of the term structure. *Journal of financial economics*, *5*(2), 177–188.

Vijayanarasimhan, S., & Grauman, K. (2014). Large-scale live active learning: Training object detectors with crawled data and crowds. *International Journal of Computer Vision*, *108*(1-2), 97–114.

Warmuth, M. K., Liao, J., Rätsch, G., Mathieson, M., Putta, S., & Lemmen, C. (2003). Active learning with support vector machines in the drug discovery process. *Journal of Chemical Information and Computer Sciences*, *43*(2), 667–673.

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning*. the MIT Press.

Wipf, D. P., & Nagarajan, S. S. (2008). A new view of automatic relevance determination. In *Advances in neural information processing systems* (pp. 1625–1632).

Wright, S. J., Nowak, R. D., & Figueiredo, M. A. (2009). Sparse reconstruction by separable approximation. *Signal Processing, IEEE Transactions on*, *57*(7), 2479–2493.

Wu, K.-P., & Wang, S.-D. (2009). Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space. *Pattern Recognition*, *42*(5), 710–717.

Wu, T. T., & Lange, K. (2008). Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 224–244.

Yang, J., Estivill-Castro, V., & Chalup, S. K. (2002). Support vector clustering

through proximity graph modelling. In *Neural information processing, 2002. iconip'02. proceedings of the 9th international conference on* (Vol. 2, pp. 898–903).

Yang, S.-H., & Lee, J. (2011). Predicting a distribution of implied volatilities for option pricing. *Expert Systems with Applications*, *38*(3), 1702–1708.

Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(1), 49–67.

Zha, Z.-J., Wang, M., Zheng, Y.-T., Yang, Y., Hong, R., & Chua, T.-S. (2012). Interactive video indexing with statistical active learning. *Multimedia, IEEE Transactions on*, *14*(1), 17–27.

Zhang, T., & Oles, F. (2000). The value of unlabeled data for classification problems. In *Proceedings of the seventeenth international conference on machine learning,(langley, p., ed.)* (pp. 1191–1198).

Zhu, X., Lafferty, J., & Ghahramani, Z. (2003). Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *Icml 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining* (pp. 58–65).

Zhu, X., Lafferty, J., & Rosenfeld, R. (2005). *Semi-supervised learning with graphs* (Unpublished doctoral dissertation). Carnegie Mellon University, Language Technologies Institute, School of Computer Science.

국문초록

빅데이터 시대가 도래하면서, 데이터 분석을 위해서는 시간과 저장 양쪽 측면에서 더욱 효율적인 알고리즘이 요구된다. 축약 학습 모델 (sparse learning model) 은 데이터의 분포를 잘 설명하는 기존 모델들의 능력을 유지시키는 동시에, 이러한 요구를 만족시킨다. 따라서, 축약 학습 모델은 2000년대 중반부터 널리 연구되었으며, 금융을 포함한 다양한 산업 분야에서 발달된 데이터 저장 기술이 적용되면서, 이러한 축약 모델이 기존에 존재하는 파라미트릭 모델에 비하여 더 정확하고 효율적인 모델을 구성할 가능성이 생겨났다.

본 연구에서는, 커널 방법론과 ARD (automatic relevance determination) 사전 분포를 가지는 두 축약 학습 모델을 개발하였다. 또한, 축약 모델과 완전 (full) 모델들을 포함한 다양한 학습 모델들을 두 가지 금융 기술과 관련된 문제에 적용하였다.

첫 번째 모델은 ARD 사전분포를 가지는 GP (Gaussian process) 회귀 모형과 가변 GP 노이즈로부터 유도된 서포트 함수를 가지는 축약 서포트 기반 군집화 모델이다. 제안된 방법론은 기존 연구에서 제안된 서포트 함수보다 더 적은 대표 벡터로 표현되는 특징을 가지고 있다. 제안된 방법론의 또 다른 특징으로는 이러한 대표 벡터들이 학습 기간 동안 자동으로 학습 데이터의 중앙에 위치하도록 선택이 되는 것이다. 다양한 군집화 문제의 시뮬레이션 결과를 통하여 제안된 방법론이 위와 같은 특징들을 이용하여 레이블링 시간을 유의미하게 감소시키는 것을 확인하였다. 또한 제안된 방법론을 이용하여 선택된 손으로 쓴 숫자들의 대표적인 형태 또한 제시되었다.

두 번째 모델은 축약 베이지안 회귀 분석을 위한 능동 학습 (active learning) 알고리즘이다. 능동 학습은 기계학습의 크고 중요한 분야 중 하나로, 만들어진 학습 모델로부터 능동적으로 선택된 상대적으로 적은 수의 레이블 된 데이터를 이용하여 정확한 학습 모델을 만드는 것을 목표로 한다. 이러한 능동 학습은 일반적으로 레이블을 얻는 비용이 큰 경우 요구된다. 본 연구에서는 두 세부 단계를 통하여 본 알고리즘을 제안하였다. 첫째로, 대표 벡터를 레이블이 된 데이터 뿐 아니라, 레이블이 되지 않은 데이터로부터도 선택하는 변환적 (transductive) 이고 일반화된 RVM (relevance vector machine) 회귀 모델을 개발하였다. 다음으로, 개발된 모델을 통하여 자동으로 선택된 대표 벡터들을 이용하여 레이블 될 데이터들을 능동적으로 선택할 수 있는 세 가지 조회 전략을 제시하였다. 제안된 방법론은 여러 인공 데이터와 실제 데이터에 적용되었고, 대부분의 경우에서 임의 선택보다 통계적으로 유의미한 결과를 나타내었다.

학습 모델을 금융 데이터에 적용함에 있어, 본 연구에서는 시장 충격 비용과 신용 부도 스왑 스프레드의 예측에 집중하였다. 첫 번째 변수인 시장 충격 비용은 기존에 학습 모델을 통하여 분석된 적이 없으며, 두 번째 변수인 신용 부도 스왑 스프레드는 소수의 연구가 진행되었으나, 그 중 다양한 최신 학습 모델을 적용하고 그들을 서로 비교하는 연구는 존재하지 않았다.

시장 충격 비용 예측의 경우, 미국 주식시장의 단일 거래 데이터에 SVR (support vector regression), RVM의 두 축약 학습 모델과, 인공신경망 (neural networks), 베이지안 인공신경망, 그리고 GP의 세 완전 모델을 적용하여 기준 파라미트릭 모델과 함께 서로의 결과를 비교하였다. 챕터 4에서 제안된 능동 학습 방법론 또한 시장 충격 비용을 예측하기 위해 적용되었다. 결과적으로, SVR을 제외한 모든 학습 모델이 파라미트릭 기준보다 좋은 성능을 보였고, 능동 학습 또한 모든 데이터를 전부 사용한 축약 베이지안 회귀 모델보다 훨씬 적은 수의 레이블 된

데이터로 임의선택보다 좋은 결과를 나타내었다.

신용 부도 스왑 스프레드 예측의 경우, 위와 같은 다섯 학습 모델과 더불어 하나의 파라미트릭 기준 모델을 기업들의 신용 위기가 매우 컸던 금융 위기 기간을 포함한 2001년부터 2014년까지의 일간 신용 부도 스왑 스프레드에 적용하고 그 결과를 비교하였다. 이번 문제에서도, SVR은 좋지 않은 결과를 나타내었으며, 특히 신용 위기가 높은 경우 더 좋지 않은 결과를 나타내었다. RVM은 SVR보다는 훨씬 좋은 결과를 보였으나, 다른 완전 모델들에 비하여는 좋지 않은 결과를 보였다.

주요어: 군집화, 능동 학습, 축약 베이지안, 금융 기술

학번: 2012-30287