11-7-2017 10:30 AM

Data-Adaptive Kernel Support Vector Machine

Xin Liu
The University of Western Ontario

Supervisor
Dr. Wenqing He
The University of Western Ontario

Graduate Program in Statistics and Actuarial Sciences
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy

Abstract

In this thesis, we propose the data-adaptive kernel Support Vector Machine (SVM), a new method with a data-driven scaling kernel function based on real data sets. This two-stage approach of kernel function scaling can enhance the accuracy of a support vector machine, especially when the data are imbalanced. Followed by the standard SVM procedure in the first stage, the proposed method locally adapts the kernel function to data locations based on the skewness of the class outcomes. In the second stage, the decision rule is constructed with the data-adaptive kernel function and is used as the classifier. This process enlarges the magnification effect directly on the Riemannian manifold within the feature space rather than the input space. The proposed data-adaptive kernel SVM technique is applied in the binary classification, and is extended to the multi-class situations when imbalance is a main concern. We conduct extensive simulation studies to assess the performance of the proposed methods, and the prostate cancer image study is employed as an illustration.

The data-adaptive kernel is further applied in feature selection process. We propose the data-adaptive kernel-penalized SVM, a new method of simultaneous feature selection and classification by penalizing data-adaptive kernels in SVMs. Instead of penalizing the standard cost function of SVMs in the usual way, the penalty will be directly added to the dual objective function that contains the data-adaptive kernel. Classification results with sparse features selected can be obtained simultaneously. Different penalty terms in the data-adaptive kernel-penalized SVM will be compared. The oracle property of the estimator is examined. We conduct extensive simulation studies to assess the performance of all the proposed methods, and employ the method on a breast cancer data set as an illustration.

KEYWORDS: Classification; Data-adaptive kernel SVMs; Imaging data; Multi-class classifier; Predictive Model; Separating hyperplane; Support vector machine.

Acknowledgments

I really appreciate the great help from my advisor, Dr. Wenqing He, during the four-year research. Not only has Dr. He always given me brilliant ideas in technical problems, but also in many other non-academic aspects, such as sharing with me much helpful experience in how to manage time efficiently, to work with others in a team and to start academic career in the future. With his encouragement, I have finished this research work. It is a great honor to work with Dr. He.

Special thanks also goes to my families. They have been supporting me against every difficulty. Every time I suffered, I turned to them. It was a hard time to separate with my families. My friends and colleagues in the department also contributed to my research. Discussions with them were pleasant and delightful.

I would like to appreciate my examination committee members, Dr. Ian McLeod, Dr. Yao Yu, Dr. Pei Yu and Dr. Ji Zhu, for their suggestive comments and inspiring recommendations. I have got many new ideas from the questions during the examination, and the experience was a great treasure to me. Also, the R developers and contributors, especially of the R package of 'e1071', have provided such a good coding environment that I can easily test my academic idea during the research.

This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada and a team grant from the Canadian Institute of Health Research, which are acknowledged.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Prediction and Classification

A prediction is a statement about an uncertain event. When decisions are to be made, the best action to achieve goals and avoid potential problems depends on a good prediction. In science, a prediction forecasts quantitatively on what will happen under specific circumstances, or connects possible causes and effects. Thus, how to make predictions accurately is of great importance in almost every discipline in science.

Specifically, the statistical learning theory provides a framework that deals with the problem of finding an optimal prediction based on data. As part of statistical inference, prediction is at the heart of almost every scientific discipline, and is one of the central topics in statistical science. Statistically, statistical learning methods aim to construct a way to describe an unknown dependency or association among measurements of objects and some characteristics from them, so that prediction of unknown information can be given based on the known information. The measurements are generally assumably easy to observe in almost all objects in which we are interested. Contrarily, characteristics of the objects may be only observable for a small part of the objects. Thus, estimating the potential association between the

input and output is of real use when the properties of the objects are invisible by observation. In other words, we try to decide the numerical characteristics of the output for any object, which is the purpose of statistical learning. Correspondingly, the estimated dependency is often referred to as a predictive model. Classification is an important constituent either for analysis of data or for prediction. In statistics, classification identifies which category a new object may belong to, based on a data set for training that contain observed objects. In other words, the goal of statistical classification is to use objects' features to identify which class they belong to, respectively. This corresponds to estimating a function which can assign correct labels to new objects based on previous observations. The basic classification problem is a binary classification issue, involving creating a decision rule so as to classify objects into one of two classes. More complicated applications have been found in multi-class categorization problem, where more than two classes are available.

Illustrative examples of classification problems can be found in different subjects. Take a real problem in medical science as an instance. Doctors need to predict whether a patient has already got a prostate cancer or not, based on data from medical images such as magnetic resonance imaging (MRI). When measurements of MRI from a patient are available on each voxel, doctors can contour the boundary of the cancered cells, by telling whether cancer has been found on each voxel, and correspondingly classify the patient as a prostate-cancer carrier or not. Or doctors can class how severe the cancer is by how much in percentage the cancered area in some organ such as a prostate takes possession of.

1.2 Motivation of the Research

Our research is partially motivated by an ongoing prostate cancer imaging study, but our method has a broad scope of application. Traditionally, areas of cancer are determined visually by examining the images of the suspected cancer areas, therefore

is not completely reliable. It is thereby desirable to develop a diagnosis process using imaging data where real correspondence between imaging and cancer is known. In this study, the images of the prostate gland are taken when the gland is in the body (in-vivo imaging data), and then the gland is surgically removed out of the body and the images of the gland are taken when it is outside the body (ex-vivo imaging data). Pathologists examine the sliced gland using high resolution microscope to identify the exact position of the cancer in the gland and a co-registration process is employed to build the correspondence between histological and imaging data in each voxel. A prediction model is expected to be constructed to predict cancer status using the in-vivo imaging data for the defined voxels, and to be utilized for diagnosis, targeted biopsy and targeted treatment in the future. In this study, the raw intensity measurements are obtained using imaging techniques such as Magnetic Resonance (MR) or Computed Tomography (CT). Usually there are around 170,000 to 200,000 voxels for each patient, with only 5% to 10% cancer voxels, which makes the cancerous and non-cancerous classes to be extremely imbalanced. A classifier that can perform well for extremely imbalanced data is then urgently needed, which promotes our consideration of support vector machines for dealing with imbalanced data.

During the first phase of the study, the binary classification for cancer at a specific voxel is expected, no matter what type of cancer or how severe the cancer is. Thus, all different labels that indicate cancer will be first classified into the cancer class, and other diseases and non-cancerous voxels are all labeled as non-cancer. The second stage is the multi-class classification based on the imaging data. During the second phase, 9 common classes are provided and listed as follows, indicating different diseases and different levels of severity of cancers.

- Atrophy: As means literally (non-cancer);
- EPE: Prostatic intraepithelial neoplasia (non-cancer);
- PIN: Prostatic intraepithelial neoplasia (non-cancer);

- G3: Tumour focus that is all gleason 3 (cancer);
- G4: Tumour focus that is all gleason 4 (cancer);
- G3+4: Tumour focus all predominately G3 with intermingled G4 (cancer);
- G4+3: Tumour focus all predominately G4 with intermingled G3 (cancer);
- G4+5: Tumour focus all predominately G4 with intermingled G5 (cancer);
- OtherProstate: Prostate tissue that does not fall into the above categories (non-cancer).

The labels of the classes are given on each voxel, indicating that there is only one label from different types of cancer on each voxel, even if it is likely to have voxels (indicating different areas of the prostate tissue) with different classes for a specific patient. A patient that has $G3+4$ type cancer in some areas is likely to have $G3$ type cancer in other areas as well as $OtherProstate$ type voxels that indicate healthy tissues. Thus, the main goal is to predict categories voxel-wisely. There are several labels that are associated with $G5$. However, the whole date set contains only one patient with very tiny area of $G5$ and associated type of cancer, without any other type of cancer. Thus, these voxels are not included in the training process, and only the sharing types of the cancer labels are to be predicted and classified.

1.3 Some Classification Methods

Many efforts have been devoted to the development of classifiers. In general, the classifiers can be sorted in two categories, namely linear and non-linear classifiers, by linear separability, i.e. separable by a linear function.

1.3.1 Linear Classifiers

Linear classifiers give labels to objects by making a decision on the basis of the value of a linear combination of different input features (Fig 1.1). The linear discriminant analysis (LDA) proposed by Fisher is the first attempt to handle the classification problems, and has been widely used to separate two or more classers of objects. Closely related to analysis of variance (ANOVA) and regression analysis, LDA works when the measurements from predictors for each observation are continuous, with the assumption of a normal distribution of error terms in the model and independence between predictors.

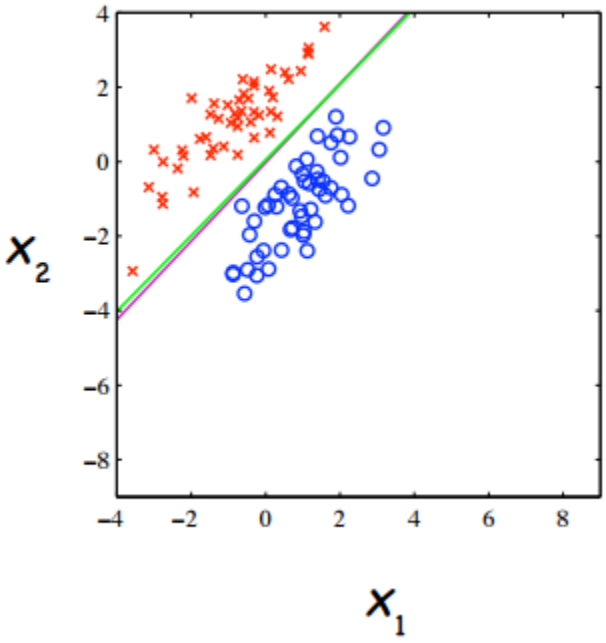

Figure 1.1: *An example of a linear classifier.*

Another popular method, the logistic regression or the logit model is derived from the linear discriminant analysis. It has been used to give an estimate of the probability of success to binary responses based on one or more predictors, and label the objects according to the probability estimated. When the regression model is built, a logistic function, or the cumulative logistic distribution function is used. Similar

techniques have been applied in the probit regression, where a cumulative normal distribution function is used instead. An illustrating example can be the relation between the chance of passing an exam and hours of studying on it (Fig 1.2).

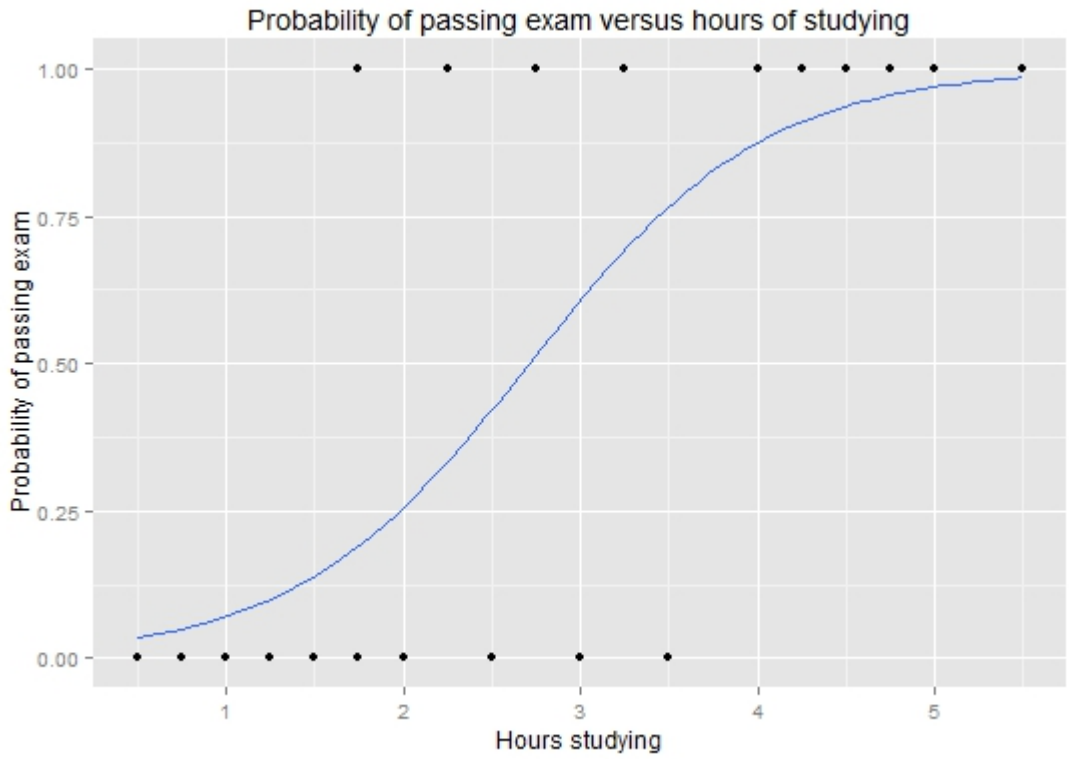

Figure 1.2: *Probability of passing an exam vs hours of studying.*

1.3.2 Non-Linear Classifiers

Due to restriction of linear separability, which is in general not available, non-linear classifiers are developed. For example, the K-nearest neighbours (KNN) is a simple but useful approach, popularly used for classification. The estimated class for an object is decided by the majority votes from its neighbouring objects, with the class of the most commonly voted ones within its K nearest neighbours, where K is a positive integer. Weights can be assigned to the neighbours to ensure that the closer neighbours have more contribution to estimation. This idea can be extended to multi-class

classification easily. K is the tuning parameter that controls the misclassification rate (Fig 1.3).

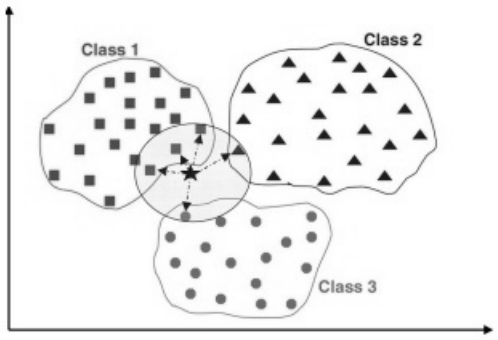

Figure 1.3: *A KNN classifier, where* $k = 5$*.(Hastie et al. (2001))*

Another one is the classification tree, a predictive model that introduces decision tree into classification problem. Leaves in the tree structure represent class labels and branches represent conjunctions of features leading to the corresponding labels. Simply understood and interpreted, the classification tree performs robustly with large data sets, while the computational cost it brings and over-complex trees under optimality may be a concern when the data are big (Fig 1.4).

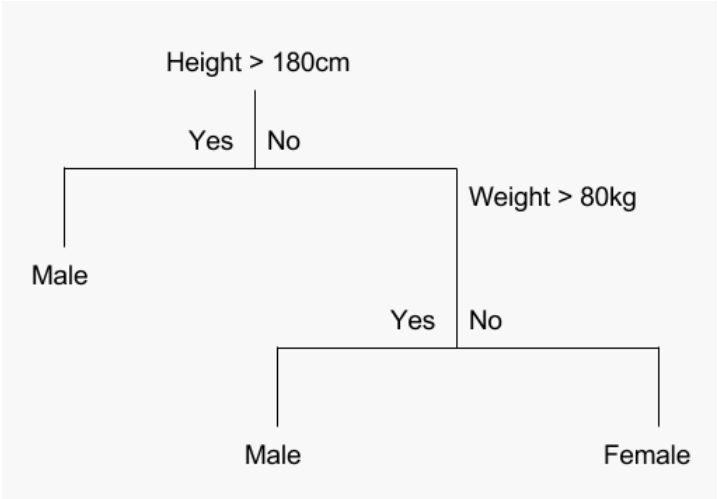

Figure 1.4: *A classification tree with 2 input variables.*

Artificial Neural Networks (ANNs) are a set of models to estimate functions with many input features. Inspired by the biological neural network, the ANN method is applied as a system of interconnected nodes, delivering messages to each other with numeric weights that are tunable by training. However, the "black-box" classification process is difficult to be interpreted (Fig 1.5).

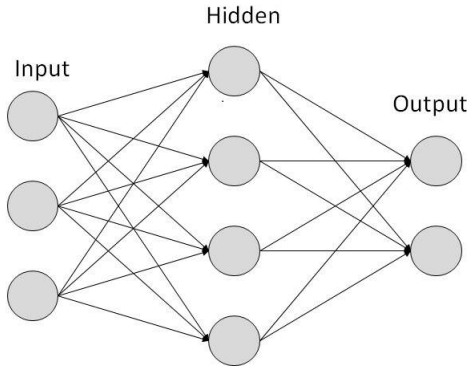

Figure 1.5: *Artificial Neural Networks.*

Although there are quite a few classifiers available either linear or non-linear, most of them bear limitations. For example, linear classifiers tend to perform unsatisfactorily when there is lack of linear separability, while non-linear ones can fix the issue in some sense. In addition, for the linear discriminant analysis and logistic regression, parametric assumptions have to be made, which may not be reasonable sometimes. Artificial neural networks are hard to be interpreted due to the "black box" effect, while the classification tree can be over-complex even though it is relatively easy to read. For the $K-$nearest neighbours method, the computational cost can be huge when the sample size is too big. Thus, how to construct an accurate and robust classifier is of great importance, particulary for imbalanced data sets.

1.4 The Support Vector Machine and Kernel Machines

The Support Vector Machine (SVM) may be the most popularly used algorithms on classification, well renowned for its strong foundations in theory, performance of generalization and the capability in high-dimensional setting. It solves the classification problem by either a linear or nonlinear separating surface. Initially proposed by Vapnik and co-authors (Vapnik and Vapnik (1998)), the SVM has been improved during the last decades a great deal, and has been increasingly important for making predictions as an essential statistical learning method. By implicitly mapping the training data into a high-dimensional 'feature space', an SVM can construct a hyperplane (a linear decision surface) within the feature space, and maximize the margins of separation between itself and the points locating closest to it. Then, this hyperplane is used as the rule to classify new objects. The idea can be easily implemented in the binary class case and extended into the multi-class case.

As an SVM allows misclassification, the trained model can map the observed objects (points in perspective of high dimension) into another space so that the objects are separated by a (probably curved) gap as wide as possible. Then unknown objects will be projected into the space created by the SVM and assigned to a specific category by which side of the gap they are located on (Fig 1.6). More recently, SVMs have been successfully applied in multi-label issues as well, where an object is possible to have more than one labels simultaneously rather than exclusively.

Mathematically, the optimization problem during training an SVM is actually controlled by a function called **kernel**, as will be explained in Chapter 2. Theoretically, it can be proved that the accuracy of an SVM relies on the selection of the kernel function. Thus, how to choose an appropriate kernel can be critical to the performance of an SVM, and the performance of an SVM may be improved by modifying the kernels.

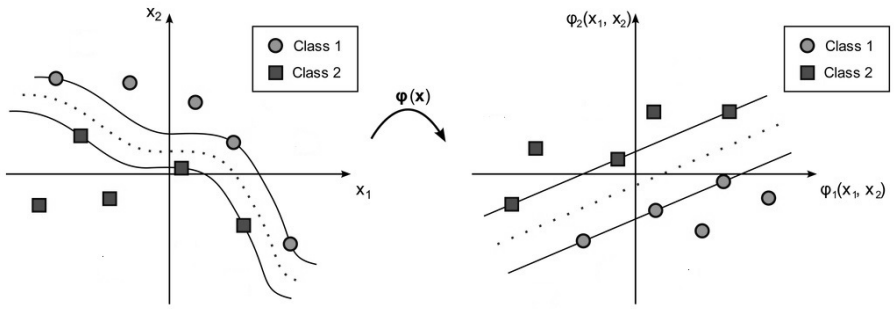

Figure 1.6: *A support vector machine classifier.*

1.5 Objectives and Main Contributions

In our research, we aim to achieve three fundamental goals. To enhance the accuracy of a support vector machine for binary classification in imbalanced data, we create a data-adaptive kernel function so that the kernel is more robust to the data especially when the data set is imbalanced, and compare the performance of the data-adaptive kernel SVM with existing classifiers in the literate. The idea of the modified SVM classifier for binary case is further extended into multi-class classification. When there are multiple classes where the imbalance problem is a main concern, the kernel functions should be made in a data-adaptive fashion so that the classifier can gain better accuracy. We investigate a simultaneous feature selection and create data-adaptive kernel-penalized SVMs to simultaneously select features that are critical in constructing the decision boundary, and classify the objects with the classifier that achieves sparseness with oracle properties in a moderate high dimensional space, especially when high level of imbalance exists.

The main contributions are as follows. To enhance an SVM's accuracy, we propose a new two-stage method of kernel function scaling. Followed by the primary SVM procedure in the first stage, the proposed method locally adapts the kernel to the locations of the data on the basis of the skewness of the class boundary, and hence,

enlarges the magnification effect directly on the Riemannian manifold within the feature space rather than the input space. By the distance measured in the feature space, the conformal transformation can make full use of the updated information in the second stage. Extensive empirical studies demonstrate that our method has excellent performance.

We extend our data-adaptive SVM construction technique to the multi-class situation when the imbalance is a main concern, based on the idea from the binary data-adaptive SVM with data-adaptive kernels. The algorithm still consists of two stages. In the first stage, a standard multi-class SVM with the indirect method is constructed so that the spatial locations of all support vectors can be found. In the second stage, the data-adaptive kernels are constructed for each SVM in the multi-class case, combining the location information of the support vectors from the first stage and the information from class sizes. By enhancing the local magnification effect, the separation of the SVMs with the data-adaptive kernels constructed is more effective and robust, with the magnification effect varying along with the density of the size of neighbours, especially for imbalanced data. Numerical studies have shown supports to the proposed method.

For the simultaneous feature selection and classification by penalizing data-adaptive kernels in SVMs, instead of penalizing the standard cost function of SVMs, the penalty will be directly added to the data-adaptive kernel function that controls the performance of an SVM, by first transforming the kernel function of the SVM and then re-conducting the SVM formulation optimization and getting the classification result with sparse features selected. Different penalty terms will be compared. The oracle property of the estimating process is examined. Iterative optimization procedure will be applied as no analytic form of the estimated coefficients can be obtained. Numerical comparisons show that the proposed model outperforms with the imbalanced data and performs as well as others when the data are balanced.

1.6 Organizations of the Work

The rest of the thesis organizes as follows. Chapter 2 will introduce the framework of the support vector machine, and the relation between an SVM and a kernel function will be described. Specific formulation settings will be described as the mathematical basis for further extension in later chapters, both for binary and multi-class problems. Based on this, the SVM classifier with a data-adaptive kernel function for the binary case will be constructed in Chapter 3, along with numerical performance comparing with other competitive classifiers in the literature. Followed the idea of the binary SVM with data-adaptive kernels, in Chapter 4, the data-adaptive SVM construction technique is extended to multi-class situation when the imbalance is a main concern. Chapter 5 introduces a new method of simultaneous feature selection and binary classification by penalizing a data-adaptive kernel in SVMs. Instead of penalizing the standard cost function of SVM, the penalty will be added to the data-adaptive kernel function directly that controls the performance of SVMs. Final conclusions will be drawn and future work will be discussed in Chapter 6.

Chapter 2

Preliminary Results of Support Vector Machines

2.1 Support Vector Machine for Binary Cases

The Support Vector Machine (SVM), an essential statistical learning method proposed by Vapnik and co-authors (Vapnik and Vapnik (1998)), has shown its excellent performance in predictive applications including handwriting pattern recognition (Cortes and Vapnik (1995)), text classification (Joachims (1998)) and image retrieval (Tong and Chang (2001)). The SVM's core idea is to map the current input space into another feature space with high dimensions on the basis of a kernel function, so that the two linearly separable classes become as far as possible (Schölkopf and Smola (2002)). It is a generalized method from the *maximal margin classifier*, an intuitive classifier, which only applies for the linearly separable classes. The development of the SVM system for binary cases is the maximal margin classifier (MMC) for linearly separable case, then support vector classifier (SVC) for linear separation with allowance of some misclassified objects, and finally the support vector machine (SVM) for non-linear separation.

2.1.1 The Maximal Margin Classifier

In a space of dimension p, a hyperplane is defined as a $p-1$ dimensional flat affine subspace. Thus, the p-dimensional setting

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p = 0$$

defines a $p-$dimensional hyperplane. When a point $\mathbf{x} = (x_1, x_2, \ldots, x_p)^T$ satisfies this equation, this point $\mathbf{x}$ lies on the hyperplane. If the left hand side (LHS) is positive, then $\mathbf{x}$ is located on one side of the hyperplane, while if it is negative, then on the other side of the hyperplane. Therefore the hyperplane separates the p-dimensional space to two parts. Thus, this hyperplane is used as a linear separating surface.

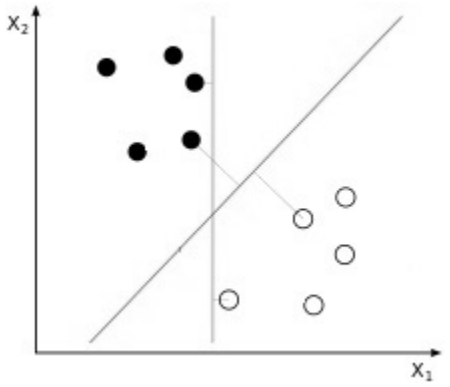

Figure 2.1: *Two possible separating hyperplanes divide two classes of observations within a 2-dimensional space.*

Suppose that a training data set contains n observations from two classes in a $p-$dimensional space, and the observations can be separated into two classes by a hyperplane perfectly based on their class labels, denoted as y, each of which takes the value of 1 or -1. For the i-th object, $y_i = 1$ when $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} > 0$, and -1 when $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} < 0$ when $y_i = -1$. Then a separating hyperplane $\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p = 0$ has the property that

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

for all the observations. When such a separating hyperplane does exist, a classifier can be naturally constructed, that is, a test object is assigned to the class based on which

side of the separating hyperplane it lies. A test object $\mathbf{x}^*$ will be classified based on the sign of $f(\mathbf{x}^*) = \beta_0 + \beta_1 x_1^* + \ldots + x_p^*$. If $f(\mathbf{x}^*)$ is positive, then the test object will be assigned to Class '1', and otherwise to Class '-1'. Besides, one may make use of the absolute value of $f(\mathbf{x}^*)$, which indicates how far the test object locates from the hyperplane. If it is much greater than zero, then the test object is located faraway from the hyperplane, and it will be confidently claimed that the classification may be accurate. Otherwise, if it is close to zero, then it might be less certain about the assignment of the class for the test object.

If such a separating hyperplane really exists, there might be infinite of them. Thus, a best one has to be decided. A natural choice is the hyperplane which is furthest from the training data so that it will be the most confident to classify the observations. As a result, the optimal separating hyperplane is the one with the maximum margin, the smallest perpendicular distance from each training observation to a specific hyperplane. This separating hyperplane is called the maximal margin hyperplane, and those training observations with equal distance from the maximal margin hyperplane are defined as the support vectors in the p-dimensional space. When these points are located differently, then the location of this maximal margin hyperplane will be changed (James et al. (2013)), in other words, the separating hyperplane is supported by these points, which are called the support vectors. However, the maximal margin hyperplane does not depend on other training observations. Even a small movement happens to any other observation, the location of the separating hyperplane will not be affected as long as it does not move across the margin boundary. This is an important property of the maximal margin hyperplane, which is the reason why the classifier is robust to some noise. Consequently, a test object can be classified on the basis of which side of the maximal margin hyperplane it is located, which is accordingly known as the Maximal Margin Classifier (MMC). The larger the margin is, the more certain it is that we are confident about which class the observation belongs to.

Denote the margin, the smallest distance between all points to the separating

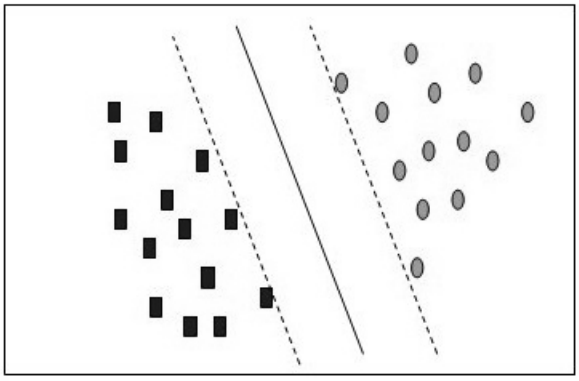

Figure 2.2: *A maximal margin classifier on two classes of observations. The solid line is the maximal margin hyperplane, with the margin as the distance from the solid line to either of the dashed lines.*

hyperplane, as G. It is straightforward that an optimal hyperplane should have the margin G as large as possible, indicating the two classes are as far as possible. Mathematically, the maximal margin hyperplane turns out to be the solution to the following optimization problem that **Maximize** G subject to

$$
\begin{aligned}
& G \geq 0, \\
& y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq G, \text{ and} \\
& \sum_{j=1}^{p} \beta_j^2 = 1, \quad \forall\ i = 1, 2, \ldots, n
\end{aligned}
$$

with respect to $\beta_0, \beta_1, \ldots, \beta_p$. The process ensures that each training data point lies on the correct side of the separating surface with some cushion value G. It is not a constraint that the sum of squares equals to one when β_i is reformed appropriately. So G can be used to represent the margin of the hyperplane and the optimization problem chooses values of β_0 to β_p to ensure that the margin G is maximized. As long as the solution to this optimization problem can be found, the maximal margin hyperplane can be obtained, and then the maximal margin classifier can do the classification.

2.1.2 Support Vector Classifiers

However, the big concern is that the maximal margin hyperplane does not always exist. Thus, it is of interest to consider a classifier which allows a slightly imperfect separating effect of the two classes, in return for more robustness to some individuals in training data and better classification of most of others. This means it is worth to give some misclassification to a few training data while the remaining data are better classified.

As a result, the Support Vector Classifier is created. Instead of finding the biggest possible margin to make every observation located on the correct side of the margin and the hyperplane, some observations are allowed to lie on the wrong side of the margin or even the hyperplane. Then, a support vector classifier will assign a test object to the class depending on which side of the hyperplane it is located. Using a support vector classifier, misclassification is allowed.

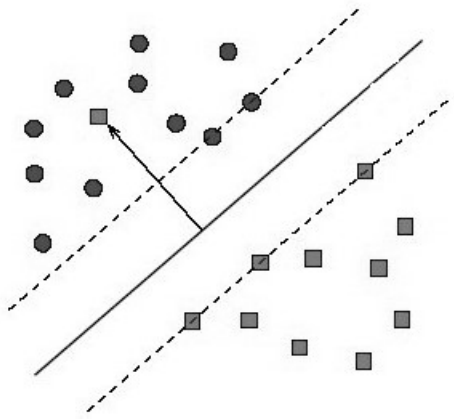

Figure 2.3: *Two classes of observations cannot be separated by any hyperplane.*

Mathematically, a support vector classifier is obtained by solving the optimization

problem: Find $\beta_0, \ldots, \beta_p, \xi_i, \ldots, \xi_n$ so that the margin G is maximized subject to

$$
\begin{aligned}
& y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq G(1 - \xi_i), \\
& \sum_{j=1}^{p} \beta_j^2 = 1, \quad \sum_{i=1}^{n} \xi_i \leq B, \\
& \xi_i \geq 0, \quad \forall i = 1, \ldots, n
\end{aligned}
$$

where $\xi_1, \ldots, \xi_n$ are known as the slack variables, allowing some individuals in training data set to locate on the incorrect side of the hyperplane, and B controls the overall tolerance of misclassification. Thus, a test object $\mathbf{x}^*$ will be classified based on the sign of $f(\mathbf{x}^*) = \beta_0 + \beta_1 x_1^* + \ldots + \beta_p x_p^*$.

To solve the support vector classifier, we need to

$$
\begin{aligned}
& \max_{\beta_0, \boldsymbol{\beta}, \boldsymbol{\xi}} G \\
& \text{subject to} \qquad y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq G(1 - \xi_i), \\
& \|\boldsymbol{\beta}\| = 1, \quad \sum_{i=1}^{n} \xi_i \leq B, \quad \xi_i \geq 0, \quad \forall i = 1, \ldots, n
\end{aligned}
$$

Note that the norm constraint on $\boldsymbol{\beta}$ can be dropped by replacing the conditions with

$$
\frac{1}{\|\boldsymbol{\beta}\|} \cdot y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq G(1 - \xi_i),
$$

(which redefines β_0) or equivalently

$$
y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq \|\boldsymbol{\beta}\| \cdot G(1 - \xi_i),
$$

Further, define $G = 1/\|\boldsymbol{\beta}\|$, and the **maximization** problem turns into

$$
\begin{aligned}
& \max_{\beta_0, \boldsymbol{\beta}, \boldsymbol{\xi}} \frac{1}{\|\boldsymbol{\beta}\|} \\
& \text{subject to} \qquad y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \\
& \sum_{i=1}^{n} \xi_i \leq B, \quad \xi_i \geq 0, \quad \forall i = 1, \ldots, n
\end{aligned}
$$

Or equivalently, solve the following **minimization** problem

$$\begin{aligned} &\min_{\beta_0,\boldsymbol{\beta},\boldsymbol{\xi}} \|\boldsymbol{\beta}\| \\ \text{subject to} \quad &y_i(\mathbf{x}_i^T\boldsymbol{\beta}+\beta_0) \geq 1-\xi_i, \\ &\sum_{i=1}^n \xi_i \leq B, \quad \xi_i \geq 0, \quad \forall i = 1,\ldots,n \end{aligned}$$

The intercept β_0 has been re-parameterized by the multiplication. Computationally, it is easier to solve the following optimization problem with penalty term, which has the equivalent solutions with the above,

$$\begin{aligned} &\min_{\beta_0,\boldsymbol{\beta},\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + B\sum_{i=1}^n \xi_i \\ \text{subject to} \quad &y_i(\mathbf{x}_i^T\boldsymbol{\beta}+\beta_0) \geq 1-\xi_i, \quad \xi_i \geq 0, \quad \forall i = 1,\ldots,n \end{aligned}$$

where the cost parameter B can replace the budget constraint on $\boldsymbol{\xi}$ (when $B \to \infty$, it corresponds to separable case). More details can be found in Hastie et al. (2001) and Tibshirani (1996). With linear inequality constraints, it is a convex optimization problem, and Lagrange multipliers method can be applied to solve the problem.

The primal Lagrange function is

$$L_p = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + B\sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i[y_i(\mathbf{x}_i^T\boldsymbol{\beta}+\beta_0) - (1-\xi_i)] - \sum_{i=1}^n \mu_i\xi_i, \tag{2.1}$$

where α_i and μ_i are Lagrange multipliers. Taking derivatives w.r.t $\beta_0, \boldsymbol{\beta}, \boldsymbol{\xi}$ and set to 0, we have

$$\begin{aligned} \boldsymbol{\beta} &= \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i, \\ 0 &= \sum_{i=1}^n \alpha_i y_i, \\ \alpha_i &= B - \mu_i, \quad \forall i, \end{aligned}$$

with the positivity constraints $\alpha_i,\ \mu_i,\ \xi_i \ \geq\ 0$ for $\forall i$. Hence, by substituting the above back into the primal objective function and re-organizing the result, we get the

Wolfe Lagrangian dual objective function

$$L_D \;=\; \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j.$$

Now, L_D needs to be maximized with the constraint $0 \leq \alpha_i \leq B$ and $\sum_{i=1}^{n} \alpha_i y_i = 0$. Additionally, the Karush-Kuhn-Tucker conditions include the following restrictions

$$\begin{aligned} \alpha_i[y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) - (1 - \xi_i)] &= 0, \\ \mu_i \xi_i &= 0, \\ y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) - (1 - \xi_i) &\geq 0, \quad \forall\, i. \end{aligned}$$

Together, these equations can give a unique solution to both of the primal and the dual problems. The solution for $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} \;=\; \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i,$$

with nonzero α_i only for those points where the constraints are exactly met, and these points are defined as the support vectors due to the fact that $\hat{\boldsymbol{\beta}}$ will be represented by them only. The decision function, or the labeling rule can be written as

$$\hat{D}(\mathbf{x}) \;=\; sign(\mathbf{x}^T \hat{\boldsymbol{\beta}} + \hat{\beta}_0)$$

Details of the optimizing the primal and dual objective functions can be found in Izmailov and Solodov (2003).

During the optimization process above, ξ_i contains the information where the $i-$th individual is located relative to the hyperplane. When ξ_i is zero, then the $i-$th individual lies on the correct side of the margin. When it is positive, then it is on the wrong side of the margin, and further when larger than 1, the wrong side of the hyperplane (Fig. 2.4).

B is a nonnegative tuning parameter, the upper bound of the sum of the $\boldsymbol{\xi}$, determining how severely the violations can be tolerated relative to the margin and

the hyperplane. When B equals to zero, which indicates no crossing to the margin, $\xi_1 = \ldots = \xi_n$ equals to zero and it goes back to the maximal margin hyperplane optimization problem. (However, there might be no solution, since a maximal margin hyperplane may not exist if the two classes are not linearly separable.) For a positive B, it is more tolerant of violations to the margin, and hence the margin may go wider and more misclassification may occur. On the contrary, when B decreases, it will be less tolerant and the margin becomes thinner. Practically, B is treated as a tuning parameter which will be selected by cross validation process.

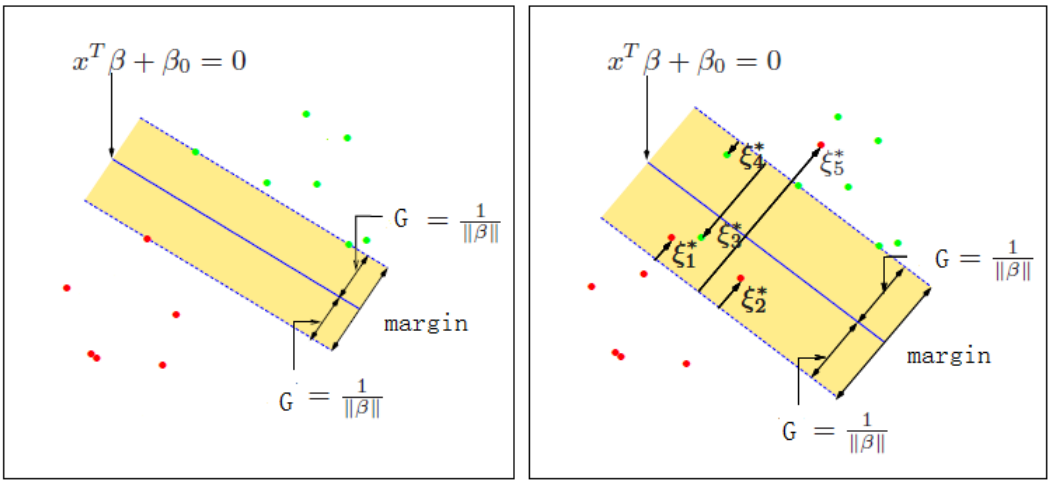

Figure 2.4: *Support Vector Classifier with Separable and Non-separable Cases. Particularly in the right panel, the points labeled ξ_j^* are on the wrong side of the margin by the amount $\xi_j^* = G\xi_j$ (Hastie et al. (2001)).*

It can be proved that only those observations locating on the wrong side of the margin may affect the location of the hyperplane (Hastie et al. (2001)). This means, those observations lying strictly on the correct side will not affect the performance of a support vector classifier, and any small movement on the location of these observations will not lead to the change of the classifier when they still stay on the correct side of the margin. Those individuals that are located on the wrong side of the margin or exactly on the margin are known as the support vectors, and they will affect the location of the hyperplane and hence the performance of a support vector classifier.

Since the support vectors are only a small part of the whole training data, the support vector classifier is very robust to those points far from the separating boundary. This is an important property different from other classification approaches such as the linear discriminant analysis or logistic regression.

2.1.3 Support Vector Machines

To make the support vector classifier with a linear boundary more attractive, it is natural to think of a non-linear surface that can be used for classification in the two-class setting. This can be achieved with more features using quadratic, cubic and higher-order polynomial transformations of the input predictors. For example, we can using $2p$ features

$$x_1, x_1^2, \ldots, x_p, x_p^2$$

where the optimization problem becomes

$$\text{maximize} \quad G$$

subject to

$$\sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2) \geq G(1 - \xi_i),$$

$$\xi_i \geq 0, \qquad \sum_{i=1}^{n} \xi_i \leq B, \qquad \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$$

However, with too many features as predictors, the classifier will become extremely complex and computationally infeasible, especially when the training observations are of limited quantity. Vapnik (2006) have shown that it is of crucial importance to make restrictions on the class of functions that can be implemented with a suitable level of complexity. In general, one can use the basis functions $\mathrm{s}_i(\mathbf{x}),\ i = 1, \ldots, l$, instead of $\mathbf{x}_i$ itself, to gain the non-linearity. Thus, if we replace $\mathbf{x}_i$ with $\mathbf{s}(\mathbf{x}_i) = (\mathrm{s}_1(\mathbf{x}_i), \mathrm{s}_2(\mathbf{x}_i), \ldots, s_l(\mathbf{x}_i))$, the classifier now will be non-linear.

Specifically, the support vector machine is an extension of the support vector classifier based on the idea by using **kernel** functions. The Lagrange dual function of the support vector machine has the form as

$$L_D \;=\; \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) >,$$

according to that for the support vector classifier in (2.1), where $<,>$ denotes the inner product. Then the solution is written as

$$D(\mathbf{x}) \;=\; \sum_{i=1}^{n} \alpha_i y_i < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) > + b,$$

and b will be solved by solving $y_i D(\mathbf{x}_i) = 1$ for any $\mathbf{x}_i$ with $0 < \alpha_i < B$. It is clear that are we only need all pairs of the inner products for training data points. Hence we represent the inner product with a general form of function as

$$K(\mathbf{x}_i, \mathbf{x}_j) \;= < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) >$$

where $K(\cdot, \cdot)$ is the so-called kernel function, quantifying the similarity of two individuals in the training data. In support vector classifier, it becomes a linear kernel function due to the linearity of the support vector classifier. When to use different forms of kernel functions in a support vector classifier, more flexible decision boundaries, which are usually non-linear, can be obtained, and the corresponding classifiers are known as the support vector machines.

2.1.4 Important Binary SVMs

One popular support vector machine model for binary cases is the least square support vector machine proposed by Suykens and Vandewalle (1999a), having discussed a version of least squares for the support vector machine classifiers. The solution proposed follows from solving a linear equation system because of the equality type of constraints in the formulae rather than the quadratic programming for the traditional support vector machines, and the entire approach is demonstrated on a benchmark classification problem with two spirals, showing a least square SVM with the radial

basis function kernel is readily found with an excellent performance of generalization and relatively low cost on computation.

Another one is proximal SVM proposed by Mangasarian and Wild (2001), known as the PSVM. Rather than assigning test objects to one of the two disjoint half hyperplanes as a standard support vector machine, the method assigns the observations to two nearest parallel hyperplanes in either input or feature space, pushing apart from each other as far as possible. The structure of the formulation can be interpreted as a regularized least squares SVM that is considered in a more general context, resulting in a extremely simple and fast algorithm that can generate a linear or non-linear classifier based merely on the solution of a single system of linear equations, much faster and more computationally economic compared with a quadratic program problem. Besides, the linear proximal SVM are proved to easily handle large data sets.

The L_1-norm SVM, proposed by Zhu et al. (2003), has the advantage over the standard L_2-norm SVM on binary-class classification, especially when the redundant noise features exist. An efficient algorithm is proposed to compute the solution path of the L_1-norm SVM, adaptively selecting the tuning parameters. The idea is further extended to multi-class classification problem.

2.2 SVMs for More than Two Classes

Originally created to separate the binary case ($k = 2$) based on the criteria of maximization on the margin, a support vector machine has been modified to discriminate more than two categories, with a wide range usage in machine learning areas such as character recognition, speech recognition, intrusion detection and bio-informatics science (Chawla et al. (2004)). Much effort has been taken to extend the binary SVM approach to the multi-class case, and a number of classifiers were proposed. There are two main methodologies to apply the SVM in the multi-class categorization problem,

namely indirect and direct methods.

2.2.1 Indirect Methods

A natural thinking of multi-category classification problem is to decompose the problem into a series of binary classifications so that the traditional SVM can be applied, which is called the indirect methods (Weston and Watkins (1999)). As two standard ensemble schemes, one-versus-one and one-versus-all are two popular techniques. Thus, to solve a $k-$class problem, at least k support vector machines have to be created, with at least k times of optimization each of which deals with a binary classification. A potential issue of the indirect methods is that each of the binary classification processes tends to become highly imbalanced along with the increasing number of categories, where the imbalanced problem during classification will occur if more sample points of a specific class than others exist. Thus, the standard SVM will be affected dramatically by the class with larger sample sizes and ignore that with smaller sizes. Consequently, the standard support vector machines will become quite sensitive to highly imbalanced classification problem due to its mechanism of construction, and will be prone to constructing classifiers which potentially have large bias to majority classes over the minority ones.

One-versus-one technique, proposed by Weston and Watkins (1998), decomposes and evaluates all potential classifiers based on the sample with k classes and hence creates $k(k-1)/2$ binary classifiers. Thus, each pairwise classifier will be applied to a test object, giving one single vote to the winning category within the two, and further the test object will be labeled to the category winning the most votes. Note that the one-versus-one approach creates many more classifiers than the number of the categories, while the size of this quadratic programming issue is smaller compared with the one-versus-all technique, making it possibly faster during the training process, and the one-versus-one method turns out to be more symmetric. Platt et al. (1999) has improved the one-versus-one technique, and further proposed a new approach

called the Directed Acyclic Graph SVM, forming a structure similar to trees so as to implement the testing process. For a $k-$class problem, the method creates $k(k-1)/2$ classifiers for every pair of classes and the bound on the test error derived relies on k as well as the margin achieved near the boundary, but not on the dimension of the input or feature space induced from the kernel, much faster and more precise compared with the traditional support vector machine.

Unlike the one-versus-one technique, the one-versus-all technique creates k individual classifiers of binary cases for a $k-$category classification problem (Weston and Watkins (1999)). Similar to an ordinary SVM process, the j-th classifier will be trained based on the observations from the $j-$th class with positive labels while the remaining $k-1$ classes as negative ones. During each process, the class of an observation will be decided by the binary classifier which offers the maximal output value, rather than the number of votes in one-versus-one technique. Thus, if all classes are supposed to have somewhat balanced sizes in training samples, the ratio of positive to negative ones for each of the k individual classifiers should be around $1/(k-1)$, indicating that the symmetry of the original classification problem is ignored.

Quite a few attempts in multi-class with indirect methods have been found in literature. For example, inspired by the idea of the least square support vector machine (LS-SVM) in binary cases, Suykens and Vandewalle (1999b) extended the method into the multi-class. A potential problem may be that the solution is created by most training sample points, referred to as the non-sparse solution. Suykens et al. (2002) further discussed an approach that overcomes the difficulties by obtaining robust estimates for predictive models leading to a weighted version of LS-SVM. The whole process is a pruning approach that can conduct pruning through physical explanation of the sorted support vectors, based on computing a Hessian matrix or its inverse. Xia and Li (2008) updated the binary LS-SVM into multi-class problem and demonstrated a sparse multi-class least square SVM where the separating boundary is determined by an optimal training data set. They proposed a variant of

binary-class least square SVM, where the solution is sparse based on the weighted coefficients of the support vectors. During the process, an adaptive regression algorithm with two stages is used to implement the training of the least square SVM, reducing the number of the parameters based on which the optimal hyperplane will be spanned.

Fung and Mangasarian (2005) followed the idea from the proximal SVM and applied the method to the multi-class classification problems. The authors proposed to balance the k classes with a novel Newton refinement modification to the proximal SVM so that the imbalanced problem associated with the one-versus-all approach can be handled. For each decomposed binary problem, the solution turns out to be similar to its corresponding binary classifier, labeling a test object to the closer class of two parallel hyperplanes torn as far as possible from each other .

Followed by the mechanism of L_1-norm SVM, Wang and Shen (2007) proposed an extended version of L_1-norm SVM on multi-class problems under the one-versus-all framework, especially when imbalanced data sets exist. The method trains the binary classifiers sequentially and treats a predictor as a relevant one for all classes if it is selected in one arbitrary binary classification process. However, when the number of the classes gets larger, each binary classification will become highly imbalanced, leading to the class with smaller fractions of sample points being ignored in non-separable cases and degrading the generalization performance. Wang and Shen (2007) has proposed L_1-norm SVM for multi-class case that can circumvent the difficulty of the one-versus-all method by treating the multiple classes in a joint way, and performs classification and variable selection process simultaneously through the sparse representation of L_1-norm.

2.2.2 Direct Methods

Another idea is proposed to straightforwardly handle the multi-class classification with a single process of optimization. These methods try to combine many binary-case

optimization processes as a single objective function, achieving category outcomes for multi-classes simultaneously (Weston and Watkins (1999) and Bredensteiner and Bennett (1999)). The main problem that comes up with the direct method is the Quadratic Programming (QP) problem, which is computationally costly due to the enormous size derived from the mechanism.

The formulation of solving the multi-class SVM problems only needs one step, and the process has variables proportional to the number of class k. Like the one-versus-all approach, it constructs k binary decision rules $D^m(\mathbf{x}), m = 1, 2, \ldots, k$, and uses the largest values of the $D^m(\mathbf{x})$ to label an test point $\mathbf{x}$. The only difference is that all the k separating boundaries are obtained by solving one single optimization problem.

Weston and Watkins (1999) have proposed a framework of the support vector machine that can enable a multi-category classification problem to be dealt with in one optimization process as well as a similar generalization of linear programming (LP) machines. Bredensteiner and Bennett (1999) have proposed a single quadratic program that can be applied in creating a non-linear classification function piecewisely, where each of the pieces can has the form of radial basis functions. Given a well-labeled training data set by $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$, where $\mathbf{x}_i \in R^p$ and $y_i \in \{1, \ldots, k\}$, the objective proposed is given as follows:

$$\begin{aligned} \underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{Min}} \quad & \frac{1}{2}\sum_{m=1}^{k}(\mathbf{w}^m)^T\mathbf{w}^m \;+\; B\sum_{i=1}^{l}\sum_{m\neq y_i}\xi_i^m \\ \text{subject to} \quad & (\mathbf{w}^{y_i})^T\mathbf{s}(\mathbf{x}_i) \;+\; b^{y_i} \geq (\mathbf{w}^m)^T\mathbf{s}(\mathbf{x}_i) + b^m + 2 - \xi_i^m, \\ & \xi_i^m \geq 0, \quad i \;=\; 1, \ldots, l, \quad m \in \{1, \ldots, k\} \end{aligned} \tag{2.2}$$

Correspondingly, the decision boundary is described by

$$\text{argmax}_m \; D^m(\mathbf{x}) \;=\; \text{argmax}_m((\mathbf{w}^m)^T\mathbf{s}(\mathbf{x}) + b^m).$$

This method has the main disadvantage of the great cost in computation from the

large size of this quadratic programming issue.

A well-known direct method in the SVM is the Crammer and Singer's multi-class kernel-based support vector machines proposed by Crammer and Singer (2001), which describes a multi-class support vector machine that can be implemented based on kernels. The start point is to generalize the notion of the margin of the multi-class problems, with which the multi-class classification problem can be casted as a restricted quadratic optimization process. Different from previous indirect methods decomposing the multiple classes problem to a sequence of independent binary classes problem, this method uses the dual of the optimization problem so that the kernels can be incorporated with a compact set of constraints and hence the dual problem can be decomposed into multiple optimization problems with reduced sizes. They described an efficient algorithm with fixed points to solve the problem.

Guermeur (2002) introduced a set of multi-class SVMs and assessed them as an ensemble of methods. The methods combine binary-case optimization problems to a single process, achieving multi-class classification simultaneously. However, the computation process is even more complex resulting from the quadratic programming issue.

Lee et al. (2004) proposed the multi-category support vector machine (MSVM) with solid theoretical properties. The proposed method shows a unifying framework with either equal or unequal classification costs. A tuning criterion for the MSVM called generalized approximate cross validation has been derived, and the effectiveness of the MSVM has been supported through applications to cancer classification with microarray data.

More recent research contributes to much more powerful modification based on the one-versus-one and one-versus-all ideas. Farquhar et al. (2006) have proposed a multi-class classifier for L_1-norm support vector machine, based on one-versus-all

idea, while a likely issue turns out to be when there are more classes, each of the binary classification problems becomes extremely imbalanced, especially when the size of a specific class is larger than that of other classes. In this scenario, the separating boundary by standard SVMs will become overwhelmed by those classes with larger scales and tend to ignore those with smaller classes. The standard SVM can perform accurately on moderate imbalanced data due to its mechanism that it is the support vectors that are used to construct the separating rules, and that the majority sample points faraway may be ignored. However, the standard SVM is quite vulnerable to highly imbalanced data case.

Followed the idea by Crammer and Singer, He et al. (2012) proposed a simple multi-class SVM with a simplified dual optimization, based on the direct classification methods. The original number of predictors in Crammer and Singer's setting, which is the product of the sample size and the number of classes, can be quite large when the sample is big. The method deals with the computation cost arising from the Crammer and Singer's method and presents the simplified multi-class support vector machine, reducing the size of the corresponding dual optimization process by introducing a relaxed bound for error during the classification process, and hence speeds up the training process without sacrificing classification accuracy.

2.2.3 Kernel Machine Methods on SVM

Quite a few kernels are created to boost the accuracy performance of an SVM. However, the optimization process is limited by the specific kernel function used, particularly when the training data set is small. When different kernel functions are employed, the performance of the classification is affected. In addition, for many other applications, such as image analysis and cancer detection (Fawcett and Provost (1997)), where the size of the training set of the target class becomes dramatically overwhelmed on the other, the separating boundary by the SVM will be skewed to the targeting class severely. Consequently, the misclassification rate can be significantly

high when the target objects are being identified (Wu and Chang (2003)).

Sánchez A (2003) illustrated the use of the kernel methods in the tasks of regression and classification and presents some latest techniques and core applications. The authors address the issues including the process of the numerical optimization, improved generalization, selection of the training set and the model, and tuning process for parameters. The application of SVMs in machine learning areas is discussed.

Tsochantaridis et al. (2004) reviewed predictive models on kernels that are positively definite. The paper describes some kernels' fundamental properties with much attention to positive definite kernel functions and their characteristics. The authors show that the kernels which are able to be written in the form of inner products agree with the class of the positive definite kernels. Sums and products of these positive definite kernels are still positive definite. Concrete examples for such kernels are given, including polynomial kernels, spline kernels, convolution kernels (specifically Gaussian and ANOVA kernels), string kernels (bag of words, n-grams, suffix trees, and mismatch kernels), locality improved kernels, tree kernels, graph kernels, kernels on sets and subspaces, and Fisher kernels. Reproducing kernel Hilbert spaces (RKHS) are discussed in the context of regularization.

Zhu and Hastie (2005) have proposed the Import Vector Machine (IVM), a new method for classification built on the Kernel Logistic Regression. The authors present that the import vector machine performs as well as the standard SVM in both binary-class and multi-class classification problems. The import vector machine gives an estimate of the probability $p(\mathbf{x}) = P(Y = 1|\mathbf{X} = \mathbf{x})$, whereas a standard SVM only predicts the sign of $[p(\mathbf{x}) - 1/2]$. The import vector machine model uses a much smaller subset of the training data (called import points) to label kernel basis functions, which may benefit the import vector machine from a possible advantage over the SVM in computation. The import vector machine model seeks a sub-model to estimate the full model created from the kernel logistic regression.

Cawley and Talbot (2007) introduced the generalized kernel machine (GKM), which is a non-linear derivations of the generalized linear model through using the trick of kernel functions. In the generalized kernel machine, a regularized generalized linear model is created within a fixed input space by a kernel, and a procedure of iteratively re-weighted least-squares is used. The regularization parameter λ and all kernel parameters are decided by minimizing an estimate of the negative log-likelihood from an efficient leave-one-out cross-validation process, and the model is selected correspondingly. Examples are presented to show the flexibility and implementation of the generalized kernel machine.

Xiang-min et al. (2007) increased separability of the training data, by proposing an adaptive similarity metric for microarray data and optimizing a re-scaled kernel function based on the kernel function. The authors demonstrate the effectiveness of the metric related to the k-nearest-neighbor classifier. A re-sampling process based on bootstrap is applied to decrease the sampling bias.

van Stiphout et al. (2011) proposed an approach of retrieving and visualizing the size of the contributions of the predictors to regression models and the way how they contribute, on the basis of visualizing trajectories with the so-called pseudo samples that represent the original predictors from the training data. Since the kernel function maps the original input space into another feature space with different dimensions, information from the predictors in the input space is not preserved. Thus, the pseudo samples are explained under the framework of the kernel partial least square regression to determine and visualize the contributions of the predictors from the input space into regression models.

Christmann and Hable (2012) address the problem of how to construct specific SVMs based on the settings of additive models, which is called additive support vector machines, and have shown that the SVMs created in such a way are consistent in

the sense of weaker assumptions compared with the standard nonparametric SVMs. Thus, the support vector machines can be applied to fit the additive models as of the form of traditional additive models. It is shown that a combination of bounded additive kernels with a Lipschitz continuous loss function can provide statistically robust support vector machines for additive models. The authors demonstrate that additive support vector machines can perform better than the standard ones when the additive model's assumptions are valid. Applications with SVMs in additive models are shown, such as quartile regression on the basis of the pinball loss function and for classification on the hinge loss function.

2.2.4 Kernel-Scaled SVM

The core idea of the SVM is to project the current input space into a feature space so that the linear separability of the classes becomes as large as possible in this feature space (Schölkopf and Smola (2002)) when in the input space the data are not linearly separable. The feature space usually has a higher dimension than the input space and it is formed based on kernel functions (Hastie et al. (2001)). Hence, it is the kernel that is crucial to determine the performance of the SVM classifier. Often, the optimal kernel function is driven by the prior knowledge of the data, and the optimization process during training the SVM is typically limited by the specific form of the kernel function that is being used, especially when the size of the training data set is small. In applications such as image analysis and cancer detection (Fawcett and Provost (1997)), the size of the training instances of the targeting class is dramatically smaller than other classes, therefore the separating boundary by the standard SVM will be skewed to the target class. In such instances, the false-negative rate will become significantly high in order to identify the target objects (Wu and Chang (2003)).

To deal with imbalanced classes in the classification problem, Amari and Wu (1999) proposed a two-stage learning procedure for choosing an optimal kernel in the support vector machine. Their idea is that the Riemannian metric will be introduced

in the feature space with a good kernel, while an ideal kernel function should have the property of enlarging the metric, consequently broadening the spatial separation between two classes. Specifically, the first stage of their method is to roughly find a separating boundary with a primary kernel function, and the kernel function is then re-scaled in the second stage using a transformation that can amplify the Riemannian metric near the separating surface found in the first stage. In other words, their algorithm finds the locations of the support vectors first by taking the fact into consideration that support vectors may be in the vicinity of the boundary. Although the performance of their classifier improves over the usual one-stage classifier in many cases, this two-stage method is vulnerable to the choice of the location of the support vectors. The location of the support vectors depends on the density region of the sample points. However, the data-dependent kernel function is constructed based on all the data in the input space, which could be inefficient and costly in computation. Although a modification of the method was proposed by Wu and Amari (2002), the algorithm still has some susceptibility, and cannot be applied in high dimensional cases.

Following Amari and Wu's idea, Williams et al. (2005) proposed a kernel scaling technique, describing a more straightforward way to achieve the useful magnifying effect. In their proposed method, the initial kernel function is transformed in a way that magnification effect will decay along with the squared distance to the separating boundary. This ensures the magnification effect to be maximized around the boundary surface and then decay smoothly to a positive constant at the margins of the region. However, their method does not take into account the imbalanced data feature, and the magnification effect decays too fast for those data not far away from the boundary according to the constant power at a specific location in the input space when the construction of the data-dependent kernel is built. Besides, the magnification effect is globally fixed, regardless of neighbors' influence, therefore can not deal with imbalanced case well.

Zhou et al. (2007) has proposed a similar idea of modifying the kernel in a data-

dependent way, by constructing the adaptive scaling function with support vectors only from the minority class, and found that the updated mapping is only associated with the adaptive scaling function and the bandwidth parameter when the primary kernel function is chosen as the Gaussian kernel function.

Maratea and Petrosino (2011) applied asymmetric kernel scaling method to imbalanced binary classification. Their basic idea is to differently enlarge areas on each side of the boundary surface so that the skewness toward minority will be compensated,by setting two free parameters in the data-dependent kernel functions in allowance for different scale levels. However, the performance does not seem to be improved much.

2.3 Simultaneous Classification and Feature Selection

An SVM offers a way of classification that has sufficient generalizing ability, fewer local minima as well as limited dependence on only a few parameters (Vapnik and Vapnik (1998)), and has achieved success in applications as a powerful classifier of high accuracy as well as flexibility. However, the method described in standard formulation settings cannot decide the importance from difference features (Maldonado and Weber (2009)), while its performance turns out to be severely affected when redundant predictors are used in deciding the separating rule, even so poor as a naive guess due to the random noises accumulated, especially in a space with higher dimensions (Hastie et al. (2001); Zhang et al. (2016)). Consequently, the development of several approaches for selecting features with SVM has been motivated, e.g. in Guyon et al. (2002); Zou (2007); Maldonado et al. (2011); Zhang et al. (2016), which provide various ways of feature ranking or selection. One of the directions, known as filter methods, filters out features with poor information based on statistical properties of features, usually done before applying any classification models. A second method,

the wrapper method, scores the whole set of features based on their predictive powers, and selects a subset of variables with the highest scores. The wrapper method shows more accuracy compared with the filter method. The most popular wrapper method for SVMs can be Recursive Feature Elimination (SVM-RFE), proposed in Guyon and Elisseeff (2003), which attempted to obtain a best subset of r features among m predictors ($r < m$), on the basis of a sequential backward selection technique. However, they all have the limitations of not taking into consideration the combination of features which optimize the performance of the classifier simultaneously.

Correspondingly, the embedded methods are created so that the selection of features can be performed during the model construction. A typical way of achieving this goal is to add some extra term which can penalize the cardinality of the chosen subset of features to the standard cost function, named as hinge loss, of the support vector machines, generally with an appropriate sparsity penalty proposed by Fan and Li (2001), a method that achieves simultaneous variable selection and output prediction. Since the standard SVM is well known to fit within the regular '*loss* + *penalty*' framework with hinge loss and L_2-norm penalty, quite a few attempts have been seen to select features for the SVM by using other forms of penalty. For example, L_1-norm penalty is applied in Bradley and Mangasarian (1998); Fumera and Roli (2002); Zhu et al. (2003); Wang et al. (2006) and Wang et al. (2008) proposed the elastic net penalty for the SVM, and the adaptive lasso penalty form was proposed to penalize the SVM; Zou and Yuan (2008) suggested a $F_\infty-$norm SVM so that groups of predictors could be selected simultaneously. In recent research, Park et al. (2012) studied the smoothly clipped absolute deviation (SCAD) proposed by Fan and Li (2001) and proved an SVM's oracle property when the number of predictors penalized by SCAD is fixed.

The aforementioned penalized feature selection methods for SVMs are all based on the original input space. However, there are possibilities that those features which have been penalized and eliminated in the input space with the above methods might

be useful in the projected feature space, and hence the classifier will lose some useful information accordingly. Actually, since the SVM projects the original input space to another feature space with higher dimensions, the performance of SVM depends directly on the kernel function, as is pointed out in, for example, Wu and Chang (2003); Williams et al. (2005); Maratea et al. (2014); thus, a natural idea is to penalize the kernel function directly, so that the features that are useful in the feature space can be selected and the classification can be achieved simultaneously.

Chapter 3

Data-Adaptive Kernel SVM in Binary-Class Case

3.1 Introduction

In this chapter, to enhance the accuracy of the SVM we propose a new two-stage method of kernel function scaling. Followed by the primary SVM procedure in the first stage, the proposed method locally adapts the kernel to the location of the data on the basis of the skewness of the class boundary, and hence, enlarges the magnification effect directly on the Riemannian manifold within the feature space rather than the input space. With the distance measured in the feature space, the conformal transformation can make full use of the updated information in the second stage. Extensive empirical studies demonstrate that our method has excellent performance.

3.2 Notation and Framework

Consider a binary classification problem where a hyperplane is expected to separate the two lasses of the response y, given sample data $\{\mathbf{x}_i, y_i\}$ for $i = 1, \ldots, n$. Here for

$i = 1, \ldots, n$, $\mathbf{x}_i$ is a vector in the input space R^p, denoted as I, and y_i represents the index of the class which takes values $+1$ or -1. However, in many cases, a hyperplane does not exist in the input space to separate completely the two classes (Hastie et al. (2001)). To get around this, an SVM method projects the input data $\mathbf{x}$ into a higher dimensional feature space R^l, denoted as F, using a nonlinear mapping function $\mathbf{s} : R^p \to R^l$ (Figure 3.1), and then searches a linear discriminant function or a hyperplane in the feature space F

$$D(\mathbf{x}) \;=\; \mathbf{w}^T\mathbf{s}(\mathbf{x}) + b \tag{3.1}$$

where $\mathbf{w}$ is an $l-$dimensional vector of parameters, $\mathbf{s}(\mathbf{x}) = (\mathrm{s}_1(\mathbf{x}), \ldots, \mathrm{s}_l(\mathbf{x}))^T$ is the $l-$dimension vector in the feature space, and b is a scalar intercept term. Hence, an individual point with observation $\mathbf{x}$ can be classified by the sign of $D(\mathbf{x})$ as long as the parameters $\mathbf{w}$ and b are determined. The boundary of the nonlinear classifier is determined by $D(\mathbf{x}) = 0$ in the input space I. Theoretically, the solution to the SVM can be obtained by maximizing the aggregated margin between the separating boundaries (Boser et al. (1992)).

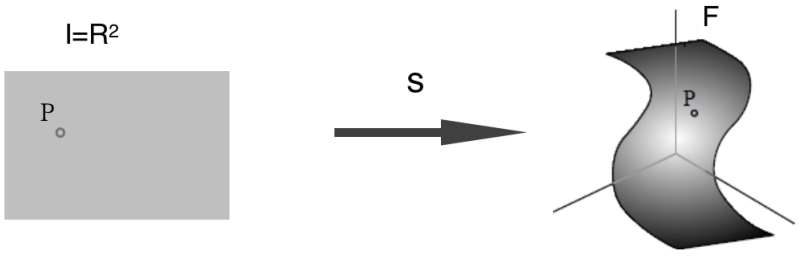

Figure 3.1: *An illustration of Riemannian manifold. By different mappings, the input space is transferred into different feature spaces (Wu and Chang (2003)).*

Mathematically, an SVM is the solution of minimizing

$$Q(\mathbf{w}, b, \xi) \;=\; \frac{1}{2}\|\mathbf{w}\|^2 + B\sum_{i=1}^{n}\xi_i \tag{3.2}$$

with respect to $\mathbf{w}$ and b, which are subject to the constraints

$$y_i(\mathbf{w}^T\mathbf{s}(\mathbf{x}_i) + b) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, n,$$

where B is the cost parameter, which determines the trade-off between the optimal combinatorial choice of the margin and the misclassification error, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^T$ is the vector of nonnegative slack variables, and $\|\cdot\|$ represents the norm. Equivalently, this optimization problem can be represented in the Lagrangian dual function with the form as

$$\underset{\boldsymbol{\alpha}}{\text{Max}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) > .$$

subject to the constraints

$$\sum_{i=1}^{n} \alpha_i y_i \;=\; 0,$$

$$0 \;\leq\; \alpha_i \leq\; B$$

for $i = 1, 2, \ldots, n$, where α_i's are the dual variables (the Lagrange Multipliers) by Lagrange Multiple Methods when solving the minimization problem in 3.2, and $< \cdot\,,\, \cdot >$ is the inner product operator. Generally a scalar function $K(\cdot, \cdot)$, which is called a kernel function, is adopted to replace the inner product of the two vectors $\mathbf{s}(\mathbf{x}_i)$ and $\mathbf{s}(\mathbf{x}_j)$ in the dual function,

$$\underset{\boldsymbol{\alpha}}{\text{Max}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{3.3}$$

Let SV be the set $\{j \mid \alpha_j > 0 \text{ for } j = 1, 2, \ldots, n\}$. Then the corresponding $\mathbf{x}_i$'s where i is in set set SV are called *support vectors*, where the cardinality of SV is l, the dimension of the feature space F. Thus, to determine the hyperplane (3.1) in the feature space, we may not need to find $\mathbf{s}(\mathbf{x})$ explicitly. We need only to find the inner products of $\mathbf{s}(\mathbf{x}_i)$ and $\mathbf{s}(\mathbf{x}_j)$, which is available from using the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$. Then the kernel form of SVM can be written as

$$D(\mathbf{x}) \;=\; \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{3.4}$$

and the estimated intercept b_j obtained by using the jth support vector $\mathbf{x}_j$ is defined as

$$b_j \;=\; y_j - \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j).$$

Hastie et al. (2001) have proved that for different j in the support vectors set SV, the b_js are the same. In practice, we can take the average of all the estimated b_j as the estimate of the bias term b.

Typical kernels in the literature include the following forms (Hastie et al. (2001)). The radial kernel has the form

$$K(\mathbf{x}, \mathbf{z}) = f(-\|\mathbf{x} - \mathbf{z}\|^2)$$

with $f(\cdot)$ being a scalar function. A popularly used radial kernel is the Gaussian Radial Basis kernel

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|^2 / 2\sigma^2),$$

where σ is the bandwidth parameter. Another popularly used kernel has the form of a polynomial function of the inner product of two vectors,

$$K(\mathbf{x}, \mathbf{z}) = f(\mathbf{x}^T \cdot \mathbf{z}),$$

where $f(\cdot)$ is the polynomial function and $(\cdot)^T$ is the transpose operator. A popular polynomial kernel with degree d has the form

$$K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x}^T \cdot \mathbf{z})^d.$$

3.3 Methodology

3.3.1 Geometric Interpretation of SVM Kernels

From the geometrical point of view, when the input space I is the Euclidean space, the Riemannian metric is then induced in the feature space F (Figure 3.2) (Wu and Amari (2002)). Denote by $\mathbf{f}$ the mapped result of $\mathbf{x} \in R^p$ in F, i.e., $\mathbf{f} = \mathbf{s}(\mathbf{x}) \in R^l$.

Then a small change in $\mathbf{x}$ in the input space, $\mathrm{d}\mathbf{x}$, will be mapped into the vector $\mathrm{d}\mathbf{f}$ in the feature space so that

$$\mathrm{d}\mathbf{f} \;=\; \nabla\mathbf{s}\cdot\mathrm{d}\mathbf{x} = \sum_j \frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\;\mathrm{d}x_j,$$

where

$$\nabla\mathbf{s} \;=\; \left(\frac{\partial\;\mathbf{s}(\mathbf{x})}{\partial\;\mathbf{x}}\right) = \begin{pmatrix} \frac{\partial\; s_1(\mathbf{x})}{\partial\; x_1} & \cdots & \frac{\partial\; s_1(\mathbf{x})}{\partial\; x_p} \\ \vdots & \vdots & \vdots \\ \frac{\partial\; s_l(\mathbf{x})}{\partial\; x_1} & \cdots & \frac{\partial\; s_l(\mathbf{x})}{\partial\; x_p} \end{pmatrix}.$$

Thus, the squared length of $\mathrm{d}\mathbf{f}$ can be written in the quadratic form as

$$\|\mathrm{d}\mathbf{f}\|^2 \;=\; (\mathrm{d}\mathbf{f})^T\cdot\mathrm{d}\mathbf{f} = (\sum_i \frac{\partial}{\partial x_i}\mathbf{s}(\mathbf{x})\;\mathrm{d}x_i)^T\cdot(\sum_j \frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\;\mathrm{d}x_j) = \sum_{ij} s_{ij}(\mathbf{x})\mathrm{d}x_i\mathrm{d}x_j,$$

where

$$\begin{aligned} \mathrm{s}_{ij}(\mathbf{x}) &= \left(\tfrac{\partial}{\partial x_i}\mathbf{s}(\mathbf{x})\right)^T\cdot\left(\tfrac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\right) \\ &= \left(\tfrac{\partial \mathrm{S}_1(\mathbf{x})}{\partial x_i},\ldots,\tfrac{\partial \mathrm{S}_l(\mathbf{x})}{\partial x_i}\right)\cdot\left(\tfrac{\partial \mathrm{S}_1(\mathbf{x})}{\partial x_j},\ldots,\tfrac{\partial \mathrm{S}_l(\mathbf{x})}{\partial x_j}\right)^T \\ &= \textstyle\sum_k \partial s_k(\mathbf{x})/\partial x_i\partial s_k(\mathbf{x})/\partial x_j. \end{aligned}$$

Consequently, the $n\times n$ matrix $S(\mathbf{x}) = [s_{ij}(\mathbf{x})]$ is defined on the Riemannian metric which can be derived from the kernel K, and $S(\mathbf{x})$ is positive definite (Amari and Wu (1999)). More straightforwardly, the following lemma demonstrates the connection between a kernel function K and a mapping $\mathbf{s}$:

Result 3.3.1. *Suppose $K(\boldsymbol{x},\boldsymbol{z})$ is a kernel function, and $\boldsymbol{s(x)}$ is the corresponding mapping in the support vector machine. Then* (3.5) *holds that*

$$s_{ij}(\boldsymbol{x}) \;=\; \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}K(\boldsymbol{x},\boldsymbol{z})|_{\boldsymbol{z}=\boldsymbol{x}}. \tag{3.5}$$

Let $v(\mathbf{x}) = \det|S(\mathbf{x})|$. The factor $\sqrt{v(\mathbf{x})}$ indicates the magnification level on the local area in F under the mapping $\mathbf{s}$, thus, is defined as the *magnification factor*. To enlarge the margin of separability between two classes, the resolution near the

boundary surface in the feature space F needs to be enlarged. This motivates us to increase the factor $\sqrt{v(\mathbf{x})}$ near the boundary of $D(\mathbf{x}) = 0$. Therefore, the mapping $\mathbf{s}$, or equivalently, the related kernel K, is to be examined so that $v(\mathbf{x})$ can be enlarged around the boundary.

For the radial kernel with the form in (3.2), it is easy to check that

$$v(\mathbf{x}) \;=\; \det|S(\mathbf{x})| \;=\; [K'(\mathbf{x},\mathbf{x})]^n \;=\; [f'(0)]^n$$

However, it is not easy to control the magnification factor $\sqrt{v(\mathbf{x})}$ by simply increasing $\mathrm{s}_{ij}(\mathbf{x})$, the entries of $S(\mathbf{x})$. Take the Gaussian kernel as an example. As pointed in later part, when K is a Gaussian kernel,

$$\mathrm{s}_{ij}(\mathbf{x}) \;=\; \frac{1}{\sigma^2} I(i=j) \tag{3.6}$$

and $\det|S(\mathbf{x})| = 1/\sigma^{2n}$. To increase the spatial resolution at a support vector $\mathbf{x}$, it is not suggested to change σ directly. On one hand, we need to accommodate the location information around the neighbourhood of the support vector $\mathbf{x}$, thus we cannot use a universal parameter σ for all these points. On the other hand, if we use different σ's for every single support vector, the parameters are too many to be tuned. Furthermore, it is not a universal way to change local resolution by only change σ, since not all radial kernel has the σ parameter; only Gaussian has. Also, the locations of the support vectors, which determine the separating boundary, need to be found. Thus, to increase the spatial resolution locally, we attempt to use the adaptive scale on the kernel function in the following part.

3.3.2 Adaptive Scale on the Kernel

To increase the spatial separability around the boundary, we propose a new two-stage adaptive scale on the kernel with two goals. One is that the spatial resolution near the margin area needs to be increased so that the separability is enhanced, while keeping the decision boundary unchanged. The other one is that the scaling process should

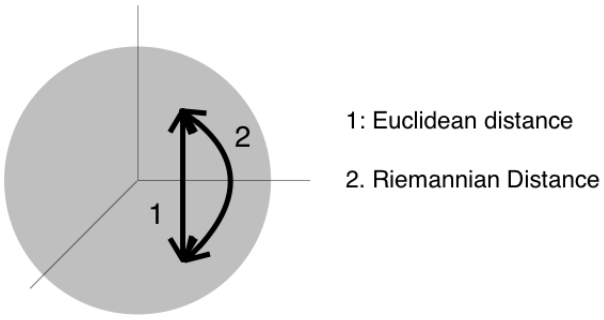

Figure 3.2: *Euclidean and Riemannian distance measures in a 3-D input space (Wu and Chang (2003)).*

only depend on the local support vectors instead of those far apart, and the scaling effect should decrease robustly and slowly with the distance approaching the boundary.

To be specific, a two-stage classification process is described as follows. We first construct a standard SVM with a primary kernel K, then we update the kernel as follows. Let $C(\mathbf{x}, \mathbf{x}')$ be a positive scalar function such that

$$C(\mathbf{x}, \mathbf{x}') \;=\; c(\mathbf{x})c(\mathbf{x}'), \tag{3.7}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are vectors from the input space, and $c(\mathbf{x})$ is a positive univariate scalar function. Then the kernel function K is updated as

$$\tilde{K}(\mathbf{x}, \mathbf{x}') \;=\; C(\mathbf{x}, \mathbf{x}')K(\mathbf{x}, \mathbf{x}') \;=\; c(\mathbf{x})K(\mathbf{x}, \mathbf{x}')c(\mathbf{x}'),$$

where $\tilde{K}(\mathbf{x}, \mathbf{x}')$ corresponds to the mapping $\tilde{\mathbf{s}}$ that satisfies the transformation

$$\tilde{\mathrm{s}}_{ij}(\mathbf{x}) \;=\; c_{ij}(\mathbf{x})\mathrm{s}_{ij}(\mathbf{x}),$$

where $c_{ij}(\mathbf{x}) = \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}C(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{x}}$. The above process is referred to as the *adaptive scaling*, and $\tilde{K}$ can be easily shown to satisfy the Mercer positivity condition, which is the sufficient condition for a real function to be a real kernel function (Wu and Amari (2002)). Thus, the metric $\tilde{\mathrm{s}}_{ij}(\mathbf{x})$, introduced from $\tilde{K}$, is related to the original $\mathrm{s}_{ij}(\mathbf{x})$ by the following theorem:

Result 3.3.2. *Given a primary kernel function $K(\boldsymbol{x}, \boldsymbol{z})$ and a scalar function $c(\boldsymbol{x})$ as in* (3.7)*, the modified mapping function $\tilde{s}_{ij}(\boldsymbol{x})$ is linked with the adaptive kernel function $\tilde{K}(\boldsymbol{x}, \boldsymbol{z})$ by*

$$\begin{aligned} \tilde{s}_{ij}(\boldsymbol{x}) &= c^2(\boldsymbol{x}) \cdot s_{ij}(\boldsymbol{x}) + c_i(\boldsymbol{x}) \cdot K(\boldsymbol{x}, \boldsymbol{x}) \cdot c_j(\boldsymbol{x}) \\ &+ c(\boldsymbol{x}) \cdot \{K_{i\cdot}(\boldsymbol{x}, \boldsymbol{x}) \cdot c_j(\boldsymbol{x}) + K_{\cdot j}(\boldsymbol{x}, \boldsymbol{x}) \cdot c_i(\boldsymbol{x})\} \end{aligned} \tag{3.8}$$

where $c_i(\boldsymbol{x}) = \partial c(\boldsymbol{x})/\partial x_i$ is the ith element of the gradient of $c(\boldsymbol{x})$, s_{ij} is determined by (3.5)*, $K_{i\cdot}(\boldsymbol{x}, \boldsymbol{x}) = \partial K(\boldsymbol{x}, \boldsymbol{z})/\partial x_i|_{\boldsymbol{z}=\boldsymbol{x}}$ and $K_{\cdot j}(\boldsymbol{x}, \boldsymbol{x}) = \partial K(\boldsymbol{x}, \boldsymbol{z})/\partial z_j|_{\boldsymbol{z}=\boldsymbol{x}}$. Particularly when $i = j$,*

$$\tilde{s}_{ii}(\boldsymbol{x}) = c^2(\boldsymbol{x}) \cdot s_{ii}(\boldsymbol{x}) + 2 \cdot c(\boldsymbol{x}) \cdot c_i(\boldsymbol{x}) \cdot K_i(\boldsymbol{x}, \boldsymbol{x}) + c_i^2(\boldsymbol{x}) \cdot K(\boldsymbol{x}, \boldsymbol{x}),$$

where $K_i(\boldsymbol{x}, \boldsymbol{x}) = K_{i\cdot}(\boldsymbol{x}, \boldsymbol{x}) = K_{\cdot j}(\boldsymbol{x}, \boldsymbol{x}) = \partial K(\boldsymbol{x}, \boldsymbol{z})/\partial x_i|_{\boldsymbol{z}=\boldsymbol{x}}$.

Detailed proof is provided in the Appendix. When $\tilde{s}_{ij}(\mathbf{x})$ has larger values at the support vectors than other data points, the updated mapping $\tilde{\mathbf{s}}$ can increase the separation when a positive function $c(\mathbf{x})$ is properly chosen. This modification of the kernel function can keep the spatial resolution stable within the feature space so that the spatial relationship between the sample points would not be changed, with $c(\mathbf{x})$ properly chosen. Also, the computational cost turns out to be quite reasonable. Inspired by the above idea, we propose to adaptively scale the primary kernel function K by constructing $c(\mathbf{x})$ with the L_1-norm radial basis function

$$c(\mathbf{x}) = e^{-|D(\mathbf{x})| \cdot k_M(\mathbf{x})} \tag{3.9}$$

and

$$k_M(\mathbf{x}) = AVG_{i \in \{j: \|\mathbf{s}(\mathbf{x}_j) - \mathbf{s}(\mathbf{x})\|^2 < M,\ y_j \neq y\}}(\|\mathbf{s}(\mathbf{x}_i) - \mathbf{s}(\mathbf{x})\|^2), \tag{3.10}$$

where $D(\mathbf{x})$ is given by (3.1), AVG denotes the average operator, y is the class label associated with $\mathbf{x}$, and M can be regarded as the distance between the nearest and the farthest support vectors from $\mathbf{s}(\mathbf{x})$.

The proposed form of $c(\mathbf{x})$ in (3.9) has many benefits. $k_M(\mathbf{x})$ reflects the spatial information of the local support vectors in the feature space F rather than the input space I. Geometrically, $k_M(\mathbf{x})$ in (3.10) is the average distance between $\mathbf{x}$ and other support vectors within a radius of M. In this way, the average value on the right hand side can comprise all the support vectors with different labels within the neighbourhood of $\mathbf{s}(\mathbf{x})$ within the radius of M. This is important when the data are imbalanced, since the globally minority class can have higher density in a small neighbourhood, making data more balanced locally. As a result, the local separability will be enhanced. By incorporating $k_M(\mathbf{x})$ into $c(\mathbf{x})$, the adaptive scaling process updates the spatial information with larger separability even when the data are imbalanced. The magnification effect is the roughly the largest near the separating boundary. It is easy to check that $c(\mathbf{x})$ gets its peak value on the separating hyperplane of $D(\mathbf{x}) = 0$, and decreases slowly to e^{-k_M} at the margins where $D(\mathbf{x}) = \pm 1$. Thus, the resolution is amplified most greatly along the boundary surface. Besides, the rate of decay from the L_1-norm is moderate compared with the standard choice such as L_2-norm, which is more robust. Thus, magnification effect still holds for the areas a bit far from the boundary surface, where the data are imbalanced locally. Additionally, if we make use of the local range tuning parameter M, the number of the support vectors to be included will be determined directly in the kernel scaling process, which adapts the classifier without too much complexity.

Our method generalizes several existing algorithms with more flexibility. For example, Amari and Wu (1999) consider the function

$$c(\mathbf{x}) \;=\; \sum_{i \in SV} e^{-k\|\mathbf{x}-\mathbf{x}_i\|^2},$$

where k is a positive scalar. Using this function, the support vectors need to locate normally near the boundary, so that the magnifying effect can be large near the support vectors and further near the boundary, as the margins are supported by the support vectors. This function can be quite sensitive to the spatial locations of the support vectors found in the first step, and thus the magnification becomes larger

at higher density regions of support vectors but drops dramatically at lower density regions. A modified function was proposed by Wu and Amari (2002), who set different k_i for different support vectors, so that the local density of support vectors can be accommodated. Though improvement of the performance is achieved by using this modified function, the computational cost becomes huge and the performance in high-dimensional data is uncertain.

On the other hand, Williams et al. (2005) uses a different $c(\mathbf{x})$ to achieve the magnifying effect. They suggested

$$c(\mathbf{x}) \;=\; e^{-kD(x)^2}$$

where $D(\mathbf{x})$ is the decision boundary and k is a positive constant for all support vectors. This function $c(\mathbf{x})$ peaks on the boundary surface $D(\mathbf{x}) = 0$ and decays to e^{-k} to the margin areas where $D(\mathbf{x}) = \pm 1$. However, the tuning parameter k is fixed throughout the whole region. This inflexibility ignores the local information. When the density of local support vectors is quite high, the separation can be inaccurate and inefficient. Furthermore, the L_2-norm decays the resolution too fast to the constant e^{-k}, which makes the separation performance unstable in high dimensional cases.

The aforementioned methods can be viewed as special cases of our proposed method. When M in (3.10) is large enough, all of the support vectors will be included in calculating $c(\mathbf{x})$, which goes back to the case of Wu and Amari (2002), and when M is sufficiently small, k_M will only depend on the local data point, without any influence from faraway support vectors, yielding the performance from method of Williams et al. (2005). By controlling the parameter M, the influence of the spatial location of the support vectors can be controlled effectively.

3.3.3 Adaptively Scaled Gaussian RBF kernel

In real applications, the primary kernel K is usually as the Gaussian radial basis function kernel

$$K(\mathbf{x}, \mathbf{z}) \;=\; e^{-\|\mathbf{x}-\mathbf{z}\|^2/2\sigma^2}.$$

Zhou et al. (2007) has shown the following results whose proofs are outlined in the Appendix.

Result 3.3.3. *If a Gaussian radial basis kernel function is adopted and a scalar function takes an arbitrary form, the modified magnification factor is*

$$\begin{aligned}\tilde{s}_{ij}(\boldsymbol{x}) \;&=\; c_i(\boldsymbol{x})c_j(\boldsymbol{x}) \;+\; c^2(\boldsymbol{x})s_{ij}(\boldsymbol{x}),\\ &=\; c_i(\boldsymbol{x})c_j(\boldsymbol{x}) \;+\; \frac{c^2(\boldsymbol{x})}{\sigma^2}I(i=j),\end{aligned}$$

where $c_i(\boldsymbol{x}) = \partial c(\boldsymbol{x})/\partial x_i$, and $I(\cdot)$ is the indicator function.

The result is quite neat in the sense that, when the primary kernel function takes the form the Gaussian RBF kernel the updated magnification factor depends only on the information from the adaptive scaling function $c(\mathbf{x})$. Thus, to make $\tilde{\mathrm{s}}$ bigger, we need to make the positive scalar $c(\mathbf{x})$ and its first order derivative, and our proposed method fulfills the purpose. When the Euclidean metric

$$\mathrm{s}_{ij}(\mathbf{x}) \;=\; \frac{1}{\sigma^2}I(i=j)$$

from (3.6) is used, the magnification factor is the constant

$$\sqrt{v(\mathbf{x})} \;=\; \frac{1}{\sigma^n}.$$

By Theorem 3.3.3 and our proposed $c(\mathbf{x})$ in (3.9), the updated magnification factor can be calculated, and the ratio of the magnification factors between the new and the old is

$$\begin{aligned}\sqrt{\frac{\tilde{v}(\mathbf{x})}{v(\mathbf{x})}} \;&=\; c^n(\mathbf{x})\sqrt{1+\sigma^2\|\nabla \log\, c(\mathbf{x})\|^2},\\ &=\; e^{-nk_M|D(\mathbf{x})|}\sqrt{1+4\sigma^2k_M^2|D(\mathbf{x})|\|\nabla|D(\mathbf{x})|\|}. \qquad (3.11)\end{aligned}$$

The ratio from (3.11) indicates that the magnification effect is almost fixed along the separating hyperplane $D(\mathbf{x}) = 0$. We can adaptively tune k_M according to the local allocation of support vectors.

3.4 Numerical Studies

To access the performance of our proposed method, we conduct extensive simulations. Also, the proposed method is compared to other methods in the literature, such as one-stage SVM, the two-stage SVM of Wu and Amari (2002) and the two-stage SVM of Williams et al. (2005). The numerical investigations made use of the software R packages. The package has mature algorithms for binary SVMs with possible arguments to provide controls of the SVM (Dimitriadou et al. (2006)).

3.4.1 Simulation Studies

We evaluate the proposed method under situations with balanced and imbalanced data. To assess the accuracy of the methods, we predict the separating boundary with the simulated sample. For all these two-stage algorithms, a Gaussian RBF kernel K is adopted in the first-stage standard SVM procedure to find approximate locations of the support vectors. Based on these, the kernel function is adaptively re-scaled to $\tilde{K}$ with corresponding $c(\mathbf{x})$ from each method, and second-stage SVM is conducted with $\tilde{K}$. Thus, we estimate a separating boundary $D(\mathbf{x})$, where $D(\mathbf{x}) = \sum_{i \in SV} \alpha_i y_i \tilde{K}(\mathbf{x}_i, \mathbf{x}) + b$. Then each sample point is assigned with a predicted label, either 1 or -1. We will assess the performance of different methods by comparing the misclassification rate.

The whole process is repeated 1000 times. In each run, the misclassification rates and their standard deviations for all methods are recorded, and the maximal margin

of error, which is defined as the largest values of the half length of the confidence interval of the misclassification rate for all compared methods, is reported as the maximal margin of error in tables. For values of the tuning parameter M, we choose the optimal value as minimizing the generalization errors by 5-fold cross validation. For the cost parameter B and the Gaussian kernel bandwidth parameter σ in the kernel function, we consider the setting of Williams et al. (2005), where the cost parameter B in (3.2) takes values 0.1, 0.2, 0.5, 1, 5, 8, 40, 100, 500 and σ takes values 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 100. We compare the classification performance by considering several sets of combinations of B and σ and report the results in the tables at the end of this chapter. During this procedure, we do not apply the cross validation procedure to find the the values of B and σ due to the computational cost. However, one can definitely apply the cross validation process to choose an optimal combination of all the parameters. The maximum test error of the naive classifier which is a random guess, is 50%.

Scenario 1: Balanced Data. In this case, the proportion of the two different classes is around 50%. Two-dimensional input data are considered as $\mathbf{x}_i = (x_{i1}, x_{i2})$ which are independently from the uniform distribution in the area of $[-1, 1] \times [-1, 1]$, where $i = 1, 2, \ldots, n$. Two classes of data are truly separated by a nonlinear curve $x_2 = \cos(\pi x_1)$ (see Figure 3.3). In other words, we let

$$y_i = \begin{cases} 1, & \text{if } x_{i2} \geq \cos(\pi x_{i1}) \\ -1, & \text{if } x_{i2} < \cos(\pi x_{i1}) \end{cases}$$

for all i from 1 to n in the sample.

It is evident that the proposed method outperforms the considered competitors. When σ gets larger with a fixed B, the misclassification rates yielded from all the methods tend to decrease. When σ is relatively small, the proposed method performs better than the methods of Wu and Amari (2002) and Williams et al. (2005); if σ is relatively large, all the methods produce nearly the same results. This is because when σ is large, the feasible solution set is large, and all the methods are capable of

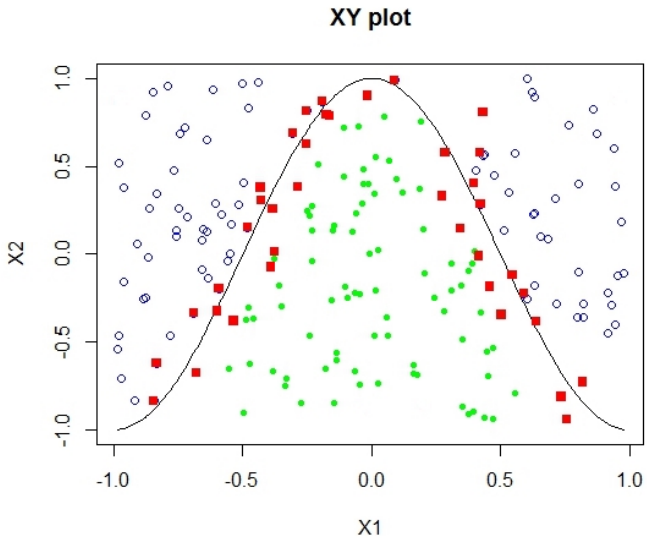

Figure 3.3: *Scenario 1: Balanced data. Label 1 is assigned to the data above the solid curve* $x_2 = cos(\pi x_1)$ *and otherwise label -1. The red dots are support vectors by the standard SVM. In this case, the proportions of the two classes are almost the same.*

finding the optimal solution. Correspondingly, when B is increasing, the budget for misclassification is getting bigger, which means more tolerance is permitted so that the two classes can be separated.

Scenario 2: Imbalanced Data.

For the case of imbalanced data, we consider that the proportions of two different labels are significantly different. Firstly, we take the proportions of the two classes as 25% versus 75%. We still choose a two-dimensional input as $\mathbf{x}_i = (x_{i1}, x_{i2})$ uniformly distributed in the area $[-1, 1] \times [-1, 1]$. The two classes are labeled by the standard normal boundary described by the function $x_2 = exp(-x_1^2)$ (see Figure 3.4), or

$$y_i = \begin{cases} 1, \text{ if } x_{i2} \geq exp(-x_{i1}^2) \\ -1, \text{ if } x_{i2} < exp(-x_{i1}^2) \end{cases}.$$

It is seen that the proportion of the hollow dot class is way much smaller than that of the solid dot class with only about 25%. The distribution of the support vectors is

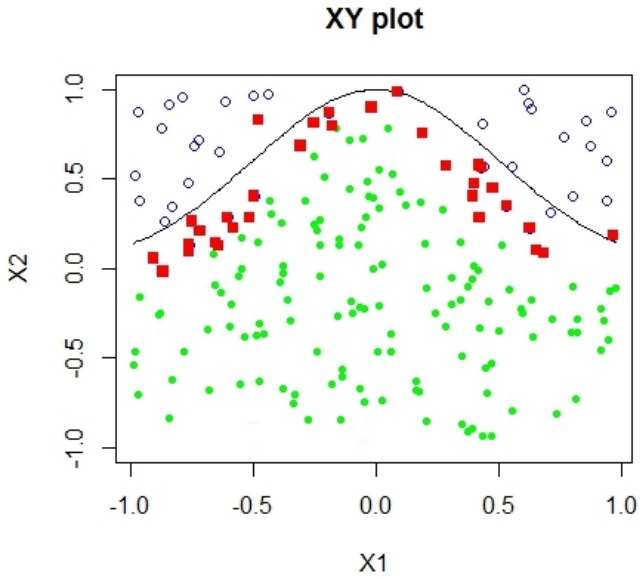

Figure 3.4: *Scenario 2: Imbalanced data with 25% and 75%. Data above the solid curve* $x_2 = exp(-x_1^2)$ *are with label 1 and below -1. The red are support vectors by the standard SVM.*

shown as the square points. B and σ are still chosen from the corresponding sets in Scenario 1. We compare the prediction performance for the combination of the cost parameter B and the Gaussian RBF kernel parameter, σ. We also apply the same procedure to a more extreme case with proportions of 10% versus 90%.

Same as in Scenario 1, we classify the data with the primary kernel K to find the locations of the support vectors of the standard SVM process, and then repeat the classification process with the proposed adaptively transformed kernel $\tilde{K}$. Different combinations of the cost B and σ are applied with 1000 times of 5-fold cross validation. The outcomes are summarised in Table 3.2 and Table 3.3.

It is seen that the performance of all the methods decays for imbalanced data due to non-uniformly distributed support vectors. The trends of misclassification rates changing with B and σ in all scenarios are similar to those in the balanced data case.

The proposed method still works the best compared to all other methods. k_M tends to change in an opposite way along with the density of the support vectors around a specific point. When the ratio of proportions become more extreme, k_M changes in a sharper way. The spatial location of the support vectors in the feature space F is therefore seen to be taken into consideration by our method.

3.4.2 Ontario Prostate Cancer MRI Data

The proposed method is also applied to a prostate cancer MRI data set arising from a cancer program of London, Ontario, Canada. This is an ongoing study conducted by the research group of Canadian Institutes of Health Research. The objective here is to classify (non)cancer areas by examining the imaging data of 21 subjects from the imaging producing equipments, such as MR, CT and ultra sound images. For each patient, 3 specific intensity measures on a voxel, namely T2W, ADC and C-Grade, coming from different platforms including MR and CT, are obtained and used as input variables. These measures range from 0 to thousands. All the 3 intensity measures are standardized, as is a usual step before further analysis, and all of them are included in the model as predictors.

We classify the data using the proposed data-adaptive scaling procedure. Specifically, two-stage SVMs are required: a first round of the standard SVM with the selected kernel is conducted, and the support vectors are obtained. The conformal transformation scalar function is then applied to update the kernel function. A second-stage SVM is then conducted based on the updated kernel and the estimated boundary is employed as the rule for classification. To choose suitable tuning parameters M, B and σ for each method, we employ the 7-fold cross validation method, where the 21 patients are randomly grouped into 7 groups with equal sizes; 6 groups of patients are used as training data, and the remaining group is used to test the error. The whole process is repeated in 1000 times.

We also analyze the data with the scaling methods of Wu and Amari (2002) and

Williams et al. (2005), the traditional SVM, and other methods including random forest and logistic regression. Training and testing error rates are reported. For adaptive scaling SVMs, the number of the support vectors is reported, while for other classifiers, the receiver operating characteristic (ROC) curves are included, and the areas under the curve are reported. They are generated on the test group of patients during the 7-fold cross validation, and the reported figures are typical ones for a specific patient.

Outcomes of classifiers with different adaptive scales are displayed in Table 3.4. By comparing the training and testing performance of the proposed method and others, we find that the proposed method generally works better than others. Support vectors obtained from the proposed method are fewer than those yielded from other methods. This can reduce the complexity in higher dimensional feature space, since the estimated decision boundary is the summation of a linear combination of the values of the kernel functions K over all the support vectors found in the first stage; when the support vectors are fewer, the values of the kernel that need to be calculated are much fewer, and hence. With other conformal transformation methods that accommodate the location of support vectors, the introduction of the tuning k_M for the location of local support vectors can greatly reduce the number of parameters that are needed to be tuned in the validation procedure.

When comparing the proposed method to other classification methods, including Logistic Regression, Random Forest, and one-stage SVM, ROC curves are given in Figure 3.5 - Figure 3.8. It is evident that the proposed method performs much better than other classifiers. For this data set, the (non)cancer labels are extremely unbalanced, and the traditional methods perform poorly, but our method performs satisfactorily.

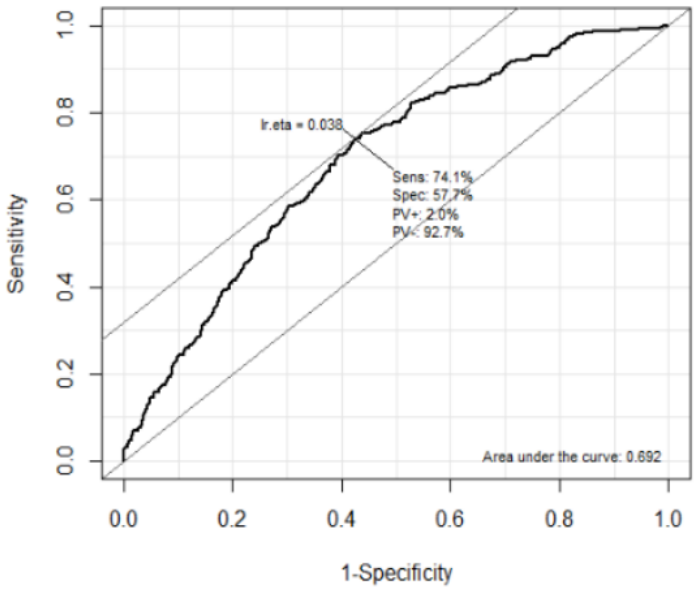

Figure 3.5: *ROC curves produced from Logistic Regression. AUC=0.692*

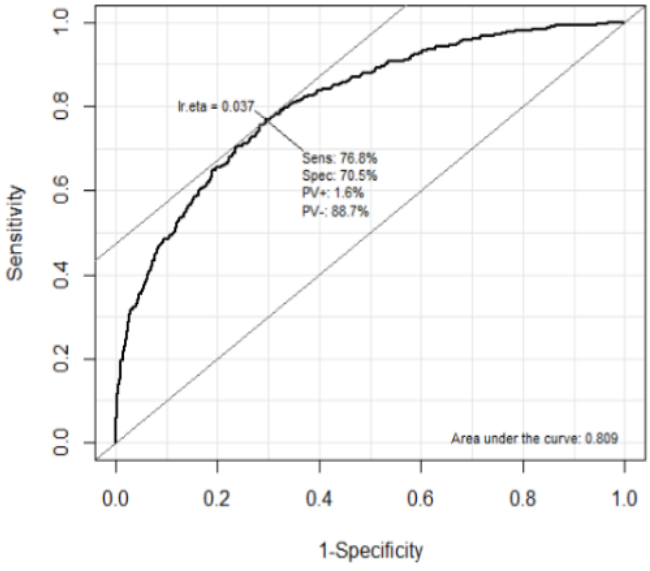

Figure 3.6: *ROC curves produced from Random Forest. AUC=0.809*

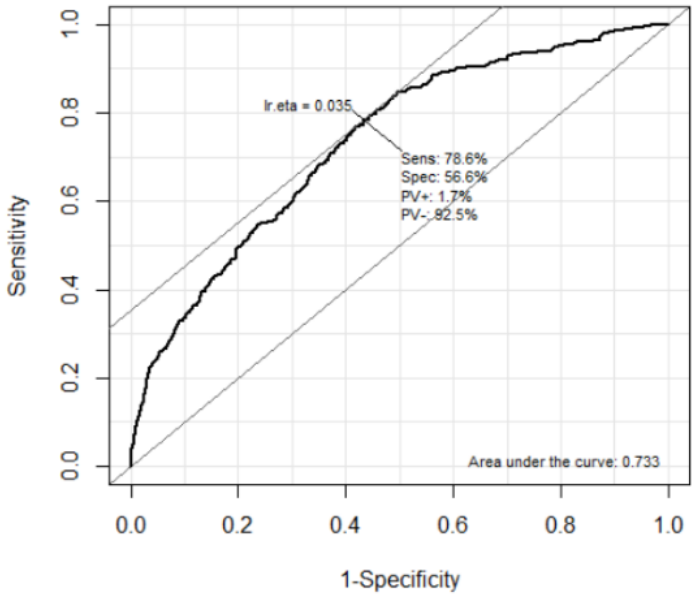

Figure 3.7: *ROC curves produced from One-stage SVM. AUC=0.733*

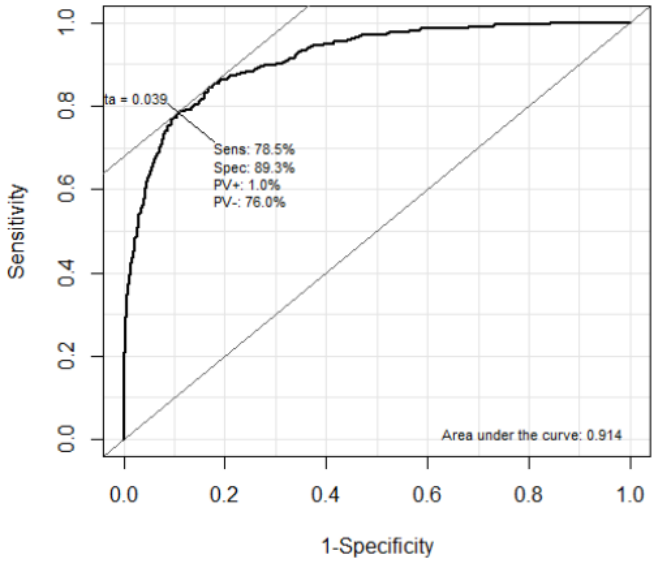

Figure 3.8: *ROC curves produced from Data-Adaptive SVM. AUC=0.914*

3.5 Concluding Remarks

In this chapter, we propose a new method of data-adaptive scaling on the kernel function in the SVM process. Our method picks the information of the local position of the support vectors to obtain a more robust solution. The model adopts the idea that the Riemannian metric in the feature space introduced by mapping with a kernel can enlarge the spatial separation between two classes, and that the locally adaptive kernel function based on the skewness of the class boundary can enhance the kernel, and hence, increase the accuracy of the classification. Simulation studies and the real data application demonstrate that our method outperform other classification methods in terms of both accuracy and robustness. Our method can be readily extended to the case with multiple classes.

To make our method more attractive, we may further pursue research in several aspects. The adaptive kernel involves two stages of the SVM procedure which are time-consuming with large data. If the spatial location of the support vectors can be approximately found and drawn with prior information, then the first round of SVM may be avoided, and those support vector candidates can be all used in the conformal transformation directly. It would be interesting to develop a more efficient algorithm. Another problem worth exploration is pertinent to the variable selection. Choosing fewer variables simultaneously in our adaptive scaling transformation can significantly reduce the model complexity, thus reducing the computational cost. Finally, when the input variables are contaminated with measurement error, which is quite common in studies such as the cancer image study, the performance of the classifier is likely to be affected. It would be interesting to investigate the measurement error effects on our method and to develop more flexible classification methods for error-prone data.

Table 3.1: *Misclassification rates for balanced data. Maximal margin of error is 0.50%.*

	One-stage SVM	Wu and Amari (2002)	Williams et al. (2005)	Proposed Method
B=8, $\sigma = 0.1$	16.30%	13.20%	12.50%	11.10%
$\sigma = 0.5$	9.60%	7.70%	6.90%	5.70%
$\sigma = 5$	7.20%	5.30%	4.70%	4.90%
B=40, $\sigma = 0.1$	18.70%	15.60%	14.10%	11.90%
$\sigma = 0.5$	15.50%	12.30%	11.10%	10.60%
$\sigma = 5$	9.30%	7.10%	6.20%	5.90%
B=100, $\sigma = 0.1$	21.10%	17.20%	15.60%	13.20%
$\sigma = 0.5$	14.00%	9.90%	8.30%	7.00%
$\sigma = 5$	12.20%	7.10%	7.80%	6.00%

Table 3.2: *Misclassification rates for imbalanced data with proportions 25% and 75%. Maximal margin of error is 0.8%.*

	One-stage SVM	Wu and Amari (2002)	Williams et al. (2005)	Proposed Methods
B=8, $\sigma = 0.1$	19.30%	15.40%	12.10%	10.50%
$\sigma = 0.5$	14.20%	12.30%	10.00%	9.50%
$\sigma = 5$	12.30%	9.60%	9.10%	7.60%
B=40, $\sigma = 0.1$	21.20%	17.70%	15.80%	13.1%
$\sigma = 0.5$	16.60%	14.60%	12.70%	11.9%
$\sigma = 5$	14.50%	11.20%	10.10%	8.20%
B=100, $\sigma = 0.1$	23.00%	18.60%	16.30%	13.90%
$\sigma = 0.5$	18.00%	16.10%	13.30%	12.10%
$\sigma = 5$	17.20%	13.60%	12.80%	8.90%

Table 3.3: *Misclassification rates for imbalanced data with proportions 10% and 90%. Maximal margin of error is 1.1%.*

	One-stage SVM	Wu and Amari (2002)	Williams et al. (2005)	Proposed Method
B=8, $\sigma = 0.1$	22.30%	18.60%	16.20%	13.30%
$\sigma = 0.5$	19.20%	15.70%	11.20%	10.90%
$\sigma = 5$	16.70%	13.20%	10.30%	8.20%
B=40, $\sigma = 0.1$	23.90%	19.60%	17.10%	14.00%
$\sigma = 0.5$	18.50%	16.50%	13.90%	12.30%
$\sigma = 5$	15.40%	12.90%	11.80%	10.10%
B=100, $\sigma = 0.1$	25.70%	19.90%	18.60%	15.10%
$\sigma = 0.5$	21.20%	17.80%	16.30%	13.50%
$\sigma = 5$	18.20%	15.20%	14.20%	11.20%

Table 3.4: *Analysis of the prostate cancer data with different methods.*

Methods	# of support vectors	Training Error	Testing Error
one-stage SVM	212	15.20%	19.20%
Wu and Amari (2002)	76	7.80%	9.50%
Williams et al. (2005)	69	7.20%	8.70%
Proposed Method	63	6.80%	7.30%

Chapter 4

Data-Adaptive Kernel SVM in Multi-Class Case

4.1 Introduction

Statistical learning from the imbalanced data turns out to be a remarkably challenging problem in multi-class cases (Menardi and Torelli (2014)). At present, many fields have seen the importance and desiring need of an accurate classifier for imbalanced data (Mazurowski et al. (2008)), including the detection of rare but serious diseases such as cancers in medical science and fraudulence issues in accounting (Chawla et al. (2004)). However, many classifiers have rather poor predictive power for the minority class and hence very likely classify most test subjects to the majority class (Maratea et al. (2014)). The main concern is the imbalance problem, which seems inevitable in the standard formulation of the SVM.

In this chapter, we extend our data-adaptive SVM construction technique to multi-class situation when the imbalance is a main concern, based on the idea from the binary data-adaptive SVM with data-adaptive kernels. The algorithm still consists of two stages. In the first stage, a standard multi-class SVM is constructed so that

the spatial locations of all the support vectors can be found. Based on this, in the second stage, the data-dependent kernels are constructed for each SVM in multi-class case, combing the spatial location of the support vectors from the first stage and the information from sample sizes. By enhancing the local magnification effect, the separation of the SVMs with the data-adaptive kernels constructed in this way is more effective and robust, with the magnification effect varying along with the density of the size of neighbours, especially for imbalanced data. Numerical studies have shown support to our proposed method.

This chapter organizes as follows. In Section 2, a brief framework of the multi-class labelling working on imbalanced statistical learning is reviewed. In Section 3, the proposed methodology is described in details that deals with the imbalanced spatial distribution of different classes. Adaptively scaling kernel function for multi-class case is proposed, involving the weights and location information for different classes, following the idea of data-adaptive kernel functions for binary case in Chapter 3. Numerical results is presented in Section 4, with comparisons to other approaches. In Section 4, concluding remarks are drawn.

4.2 Notation and Framework

4.2.1 Framework of an SVM

As a popular method for classification problem proposed by Vapnik and Vapnik (1998), the support vector machine essentially uses a kernel function that maps the original input space into a high-dimensional feature space to separate the observations with two classes as faraway as possible, preferably with a linear boundary. In a binary setting with some sample $\{\mathbf{x}_i, y_i\}$ for $i = 1, \ldots, n$, where $\mathbf{x}_i$ is a vector in the input space $I = R^p$ and y_i represents the class index which takes values $+1$ or -1, a nonlinear support vector machine maps the input data $\mathbf{x}$ into another high-dimension

feature space, $F = R^l$, using a nonlinear mapping functions$\boldsymbol{s} : R^p \rightarrow R^l$. Then a linear discriminant function

$$D(\mathbf{x}) \;=\; \mathbf{w}^T \mathbf{s}(\mathbf{x}) + b \tag{4.1}$$

will be searched for in the feature space F, where $\mathbf{w}$ is an $\ell-$dimensional vector, $\boldsymbol{s}(\mathbf{x}) = (s_1(\mathbf{x}), \ldots, s_l(\mathbf{x}))^T$ is the corresponding vector by using the nonlinear $\ell-$dimension vector function $\boldsymbol{s}(\mathbf{x})$ and b is an intercept. Hence an individual point $\mathbf{x}$ is classified by the sign of $D(\mathbf{x})$ as long as the parameters $\mathbf{w}$ and b are determined. It is obvious that the boundary of the nonlinear classifier is $D(\mathbf{x}) = 0$ in the input space I. Then, with mathematical transformation shown in Chapter 3, the kernel form of SVM can be written as

$$D(\mathbf{x}) \;=\; \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{4.2}$$

where α_i is the positive numeric derived as the dual variables by Lagrange Multiple Methods, and SV is the set of support vector indices. The intercept b can be obtained with any support vector $\mathbf{x}_j$ as

$$b_j \;=\; y_j - \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j),$$

Instead of finding the mapping pattern $\mathbf{s(x)}$ explicitly, which is not necessary, only the inner product is required, directly available from the kernel K. Still, we mainly consider the radial basis kernel of the form

$$K(\mathbf{x}, \mathbf{z}) = f(-\|\mathbf{x} - \mathbf{z}\|^2).$$

4.2.2 Popular Multi-class SVM Algorithms

Consider we classify a sample k categories. Basically, two methods, the indirect and the direct methods, can be applied, depending on whether the classification process can be achieved by one or more optimization problem. The indirect method is to construct and combine several binary classifiers, while the direct method is to directly consider all the data in one single optimization formulation. In our method, we mainly base on the one-versus-all approach due to less computational cost, and

flexibility with the data-adaptive kernel transformation.

The one-versus-all approach creates k binary SVM classifiers. The $i-$th SVM is trained with all points in the $i-$th class with positive sign '+1' while all other point labeled as negative sign '-1'. After solving the k optimization problem, there will be k decision functions constructed, and a test object $\mathbf{x}$ will be assigned to the class with the largest value of the decision function $D^m(\mathbf{x}),\ m = 1, \ldots, k$. This indicates that the test object is labeled as the class that is farthest.

However, the principal concern for multi-category classification is the imbalance issue for both indirect and direct methods. When the size of one specific category is quite small, the proportion of each binary classification problem that contains this class is imbalanced, and the classification accuracy will be affected, as has been pointed out in Chapter 3. This issue is particularly common for the one-versus-all and direct approaches, since the training data from a particular class will be separated from those from all the other classes. Another issue is the computational cost, especially for the one-versus-one approach. A $k-$class problem needs $k(k-1)/2$ kernel functions, so there will be too many kernel functions to be trained when k is large, compared with the one-versus-all and direct approach. Besides, if there is a tie in the voting process in the one-versus-one approach, the random guess may deteriorate the performance of the classification, while the other two methods can hardly see a tie issue.

As has discussed in Chapter 3, the imbalance in the binary classification can be dealt with by magnifying the resolution locally with the data-adaptive scaling on the kernel function. Thus, we can apply the method on the one-versus-all approach on each of the k binary classification processes, and update the kernel function so that the magnification effect around the separating boundary can be enhanced.

4.3 Methodology

4.3.1 Data-Adaptive Kernel SVM for the Multi-class Case

Motivated by the binary data-adaptive kernel SVM, we propose multi-class data-adaptive kernel SVM algorithm, a new way of multi-category classification problem. The main idea is to enhance the magnification effect around the multiple separating boundaries in the feature space, by adaptively scaling the primary kernel functions. The algorithm consists of two stages. During the first stage, the standard one-versus-all approach is applied and the support vectors are found based on the primarily assumed kernel functions, so that the decision functions are estimated. Based on this, the kernel functions will be updated with the spatial location information. In the second stage, the multi-class SVMs are solved with the updated kernel function, and the estimated decision functions from the second stage are used to make predictions for test objects in future.

Consider a size n training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i$ is a vector from the input space $I = R^p$, and $y_i \in \{1, 2, \ldots, k\}$, the $m-$th SVM solves the following problem:

$$\begin{aligned}
\underset{\mathbf{w}^m, b^m, \boldsymbol{\xi}^m}{\text{Min}} \quad & \frac{1}{2}(\mathbf{w}^m)^T \mathbf{w}^m \; + \; B \sum_{i=1}^{n} \xi_i^m \\
\text{subject to} \quad & (\mathbf{w}^m)^T \mathbf{s}(\mathbf{x}_i) \; + \; b^m \geq 1 - \xi_i^m, \text{ if } y_i = m, \\
& (\mathbf{w}^m)^T \mathbf{s}(\mathbf{x}_i) \; + \; b^m \leq -1 + \xi_i^m, \text{ if } y_i \neq m, \\
& \xi_i^m \geq 0, i \; = \; 1, \ldots, l,
\end{aligned}$$

where B is the cost parameter, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^T$ is the slack variable, and $\mathbf{s}$ is the mapping from the original input space to the feature space $F = R^l$. Similar to the binary case, the minimization problem has the corresponding Lagrangian dual formulation as

$$\underset{\boldsymbol{\alpha}^m}{\text{Max}} \quad \sum_{i=1}^{n} \alpha_i^m - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^m \alpha_j^m y_i y_j < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) > . \tag{4.3}$$

subject to the constraints

$$\sum_{i=1}^{n} \alpha_i^m y_i \;=\; 0,$$
$$0 \;\leq\; \alpha_i^m \leq\; B$$

for $i = 1, 2, \ldots, n$ and $m = 1, 2, \ldots, k$. α_i^m's are the dual variables (the Lagrange Multipliers) by Lagrange Multiple Methods when solving the minimization problem corresponding to the $m-$th SVM classifier, and $< \cdot\ ,\ \cdot >$ is the inner product operator. By replacing the inner product in (4.3) with the kernel function $K(\cdot, \cdot)$, the maximization problem becomes

$$\underset{\boldsymbol{\alpha}^m}{\text{Max}} \quad \sum_{i=1}^{n} \alpha_i^m - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^m \alpha_j^m y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{4.4}$$

subject to the constraints

$$\sum_{i=1}^{n} \alpha_i^m y_i \;=\; 0,$$
$$0 \;\leq\; \alpha_i^m \leq\; B.$$

Let SV^m be the set $\{j \mid \alpha_j^m > 0 \text{ for } j = 1, 2, \ldots, n\}$, containing the index of the support vectors for the $m-$th SVM. After solving the whole (4.4), there are k decision boundary functions:

$$D^m(\mathbf{x}) \;=\; \sum_{i \in SV^m} \alpha_i^m y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{4.5}$$

and the estimated intercept b_j obtained with the jth support vector $\mathbf{x}_j$ is defined as

$$b_j^m \;=\; y_j - \sum_{i \in SV^m} \alpha_i^m y_i K(\mathbf{x}_i, \mathbf{x}_j), \tag{4.6}$$

From the geometrical point of view, when the input space I is the Euclidean space, the Riemannian metric is induced in the feature space F (Wu and Amari (2002)). As has been proved in Lemma 3.3.1, the mapping $\mathbf{s}$ is actually controlled by the chosen kernel function, and the magnification effect in a neighbourhood of a sample point $\mathbf{x}$ is determined correspondingly. Thus, to increase the magnification effect, in our

previous chapter, we have proposed the data-adaptive kernel SVM, by updating the kernel function with the scaling transformation function

$$\tilde{K}(\mathbf{x}, \mathbf{x}') \;=\; c(\mathbf{x})K(\mathbf{x}, \mathbf{x}')c(\mathbf{x}'),$$

where $c(\cdot)$ is a positive scalar function,

$$c(\mathbf{x}) \;=\; e^{-|D(\mathbf{x})|\cdot k_M(\mathbf{x})}$$

and

$$k_M(\mathbf{x}) \;=\; AVG_{i\in\{j:\|\mathbf{s}(\mathbf{x}_j)-\mathbf{s}(\mathbf{x})\|^2<M,\ y_j\neq y\}}(\|\mathbf{s}(\mathbf{x}_i)-\mathbf{s}(\mathbf{x})\|^2),$$

In this way, the average value on the right-hand side can comprise all the support vectors within the neighborhood of $\mathbf{s}(\mathbf{x})$ with the radius of M but with a different label of class. This takes the spatial location of the support vectors in the feature space F into account. The method turns out to be more robust and efficient, and we adopt the similar idea to the multi-class case.

We split the sample into k categories, each of which is represented by $C_1, C_2, \ldots, C_k$ respectively, by their predicted labels of classes from the first round SVM. Then, the multi-class adaptive data-dependent kernel transformation function is proposed as

$$c(\mathbf{x}) = \begin{cases} \exp(-k_{M,1}(\mathbf{x})|D^1(\mathbf{x})|), & \text{if } \mathbf{x}\in C_1 \\ \exp(-k_{M,2}(\mathbf{x})|D^2(\mathbf{x})|), & \text{if } \mathbf{x}\in C_2 \\ \ldots & \\ \exp(-k_{M,k}(\mathbf{x})|D^k(\mathbf{x})|), & \text{if } \mathbf{x}\in C_k \end{cases} \tag{4.7}$$

where $D^m(\mathbf{x}),\ m=1,2,\ldots,k$ is given by (4.9), $k_{M,i}(\mathbf{x}),\ i=1,\ldots,k$ are parameters that will be calculated to control the decay rates, similar to $k_M(\mathbf{x})$ in (4.3.1). In terms of imbalanced data set, selection of an appropriate transformation function for each category is important so that the problem can be transferred back to the balanced one. Thus, we propose $k_{M,i}(\mathbf{x})$ is constructed as

$$k_{M,i}(\mathbf{x}) \;=\; AVG_{i\in\{j:\|\mathbf{s}(\mathbf{x}_j)-\mathbf{s}(\mathbf{x})\|^2<M\cdot w_i,\ y_i\neq y\}}(\|\mathbf{s}(\mathbf{x}_i)-\mathbf{s}(\mathbf{x})\|^2)$$

where AVG denotes the average operator, y is the class label associated with $\mathbf{x}$, and M can be regarded as the distance between the nearest and the farthest support vectors from $\mathbf{s}(\mathbf{x})$. The weighting factor w_i is defined as

$$w_i = \frac{1/n_i^2}{\sum_{i=1}^{k} 1/n_i^2}.$$

where n_i is the size of the class from the training sample size. In this way, w_i's show the sparse location nature of each category, and it is obvious that all weighting factors are summed to 1.

Here are some remarks. Controlling parameter M in binary case is replaced by $M \cdot w_i$, instead of a direct setting of M_i for $i = 1, 2, \ldots, k$. The reason is that in this way, the local spatial information and the imbalance from the sample proportions can be separated. An universal control on the Riemannian distance is adopted, instead of different M_is for different categories, while taking the weight factors into account. In this way, the classification can be more robust to extreme points in spatial locations, which will drag the classifiers towards the majority classes, while the weights are considered to somewhat balance the training sample. Beside, this choice can simplify the estimation process by significantly reducing the number of parameters to be tuned in real application. The only parameter to be tuned is M, since the weights factors can be obtained from the class sizes. This can avoid over-parameterized and over-fitted situation. Further, the weight factor has the form of the reciprocal of the squared sample size. This gives the benefits that the minority class can be updated with a larger magnification effect, and that the squared form can even enlarge this magnification.

Other scaling transformation for multi-class SVMs has been seen in literature. In terms of choice of $c(\cdot)$, there are other choices available, though they all bear some drawbacks. More details can be found in Chapter 3. There are some different types of $k_{M,i}$ parameters. For example, Zhang et al. (2014) has proposed

$$k_{M,i}(\mathbf{x}) = w_i \cdot k_i(\mathbf{x})$$

where the weight factor is defined as

$$w_i \;=\; \frac{1/n_i}{\sum_{i=1}^{k} 1/n_i}$$

and k_i is chosen according to a chi-square distribution. The problem is that, on one hand, the weight factor is not strong enough so that the magnification effect may not be sufficient; on the other hand, the chi-square adopted has limited connection during the process. Maratea et al. (2014) has the similar idea of construction of $c(\cdot)$ but still too many parameters are involved with limited improvement.

4.3.2 Data-adaptive SVM algorithm for Multi-class Case

With $c(\mathbf{x})$ constructed (4.7), we conformally transfer the kernel obtained from the first around multi-class SVM in the following

$$\tilde{K}(\mathbf{x}, \mathbf{z}) \;=\; c(\mathbf{x}) \cdot K(\mathbf{x}, \mathbf{z}) \cdot c(\mathbf{z}).$$

Note that the kernels in the first stage are all Gaussian RBF kernels. We now use the data dependent kernels $\tilde{K}(\cdot, \cdot)$ to conduct the second round SVM as

$$\underset{\boldsymbol{\alpha}^m}{\text{Max}} \quad \sum_{i=1}^{n} \alpha_i^m - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^m \alpha_j^m y_i y_j \tilde{K}(\mathbf{x}_i, \mathbf{x}_j). \tag{4.8}$$

subject to the constraints

$$\sum_{i=1}^{n} \alpha_i^m y_i \;=\; 0,$$

$$0 \;\leq\; \alpha_i^m \leq\; B.$$

The k decision boundary functions can be constructed

$$\tilde{D}^m(\mathbf{x}) \;=\; \sum_{i \in SV^m} \alpha_i^m y_i \tilde{K}(\mathbf{x}_i, \mathbf{x}) + b, \tag{4.9}$$

and we can use these decision functions to predict the labels of the test objects $\mathbf{x}$ by assigning it to the class with the largest absolute values of $\tilde{D}^m(\mathbf{x})$ for $m = 1, 2, \ldots, k$. In this way, it has all the benefits from the adaptive scaling for the binary case. Thus,

as long as the controlling parameter M and the Gaussian kernel parameters B and σ are tuned adaptively with data, the classifier can be estimated, and hence the subjects' labels can be predicted with the trained classifier.

To conclude the section, the algorithm of the whole procedure of the multi-label classification problem is described as follows. A regular SVM classifier is trained with an ordinary Gaussian RBF kernel function, and the support vectors can be found, so that the separating boundaries can be approximately found. Based on the spatial information of these support vectors, the conformal transformation will be constructed, and the original kernel function is updated. A second round of SVM optimization problem is conducted with the updated kernel function, so that the boundaries for different classes can be found. Accordingly, the predicted labels for subjects can be estimated. Performance of the method will be given in experiment results section.

4.4 Numerical Investigation

In this section, the details of how to conduct our proposed approach in real situation is described in order to assess the classification capabilities and compare with other classifiers. The whole study will be divided into two parts on two data sets, namely an artificial data set and a real data set with prostate cancer. We will compare with 4 methods from the traditional SVM, Wu and Amari (2002); Williams et al. (2005); Maratea et al. (2014) and our proposed method.

In terms of classifiers' performance, the classification results can provide quantitative measures. Generally speaking, one of the main indices is the overall accuracy. Basically, there will be four possible categories, TP (true positive), FN (false negative), FP (false positive) and TN (true negative). Hence, the overall accuracy rate can be evaluated as

$$P_{overall} = \frac{TP + TN}{TP + FP + TN + FN} .$$

However, for imbalanced data, the overall accuracy rate is not sufficient and useful in some cases (Maratea et al. (2014)). Consequently, another two measurements on classifiers' performance that are critical for imbalanced data are adopted, $F-measure$ and $G-mean$ (geometric mean), derived from the four possible outcomes as well.

Normally for classification problems, the sensitivity P_{sen} and specificity P_{spe}, which show the performance of the positive class and that of the negative class, respectively, defined as

$$P_{sen} = \frac{TP}{TP + FN}$$
$$P_{spe} = \frac{TN}{TN + FP}$$

Precision is useful, defined as the proportion of relevant cases that is defined as

$$P_{pre} = \frac{TP}{TP + FP}.$$

$F-measure$ considers both precision and sensitivity, which can be further interpreted as a weighted average of the two as

$$F - measure = \frac{2 \times P_{pre} \times P_{spe}}{P_{pre} + P_{sen}}.$$

$G-mean$ is constructed as the product of the sensitivity and the specificity, giving a more fair comparison between the positive and negative classes, regardless of its size.

$$G_{mean} = \sqrt{P_{sen} \times P_{spe}}.$$

Both the simulation study and the real data analysis will be based on these measures.

4.4.1 Simulation Studies

To start with, we consider an artificial data set. The whole process will be intensive. There will be three scenarios, each of which considers the balanced, moderately imbalanced and extremely imbalanced cases, respectively. The whole process is based

on the Gaussian RBF kernel during the first round of classification, if not mentioned elsewhere.

For convenient reasons, the input space will be 2-dimensional, and all training data will come from only 3 classes, all of which are bivariate Gaussian distributions with different but fixed means vectors as $(2,2),(4,3)$ and $(3,2)$, and equal co-variance matrix $\Sigma_{2\times 2}$, set as $\gamma \cdot \Sigma$, where γ is a controlling parameter that can control the overlapping proportion and hence misclassification will occur. Moderate covariance is allowed for all pairs as well, set as correlation efficient $\rho = 0.3$. It is worth noted that the misclassification rate will be definitely affected by the distance of the mean vectors, as it is not difficult to imagine, for instance, that the misclassification will not occur if the centres are sufficiently far from each when the covariance matrix is set as I.

The whole process is repeated 1000 times. During each time, the corresponding measures and their standard deviations for all methods will be recorded, and the maximal margin of error, which is the largest values of the half length of the confidence interval of the measures for all compared methods, is reported as the maximal margin in tables. The overall sample size for the training data will be set as 600, and will be separated into 3 class by different weights in 3 different scenarios, with the class size as $(200,200,200)$ in Scenario 1, $(100,200,300)$ in Scenario 2 and $(20,100,480)$ in Scenario 3. In each scenario, different combinations of the budget and width parameters B and σ need to be determined for the Gaussian kernel function. The cost parameter B in (3.2) takes values 0.1, 0.2, 0.5, 1, 5, 8, 40, 100, 500 and σ takes values 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 100. For values of the tuning parameter M, we choose the optimal value as minimizing the generalization errors by 5-fold cross validation.

The classification procedure will be as follows. We train the classifier with the traditional SVM, and the support vectors can be found approximately. Then, the kernel functions for all the methods will be updated adaptively by conformal transformation with different scalar function $c(\mathbf{x})$. A second round of SVM will be conducted, and

the estimated class for each observations in the sample will be given and compared with the true label. With the accuracy measures mentioned above, the performance of different classifiers will be obtained in Table 4.1, 4.2, and 4.3.

Improvement can be found in almost all situations with different combinations of the cost B and σ in the ordinary SVM. In general, the proposed method outperforms among all the classifiers, especially in the imbalanced data. When σ gets larger with fixed B, the misclassification rate tends to decrease in all of the methods compared. When σ is relatively small, the proposed methods beat Wu and Williams' methods both, while if σ is relatively larger, all of the methods are nearly the same. This is because when σ is getting larger, the feasible solution set is getting larger, and all of the methods tend to find the optimal solution. Correspondingly, when B is increasing, the budget for misclassification is getting bigger, which means more tolerance is permitted so that the two classes can be separated.

For imbalanced data the performances of all methods are a bit worse without surprise due to the non-uniformly distributed support vectors. The trends how misclassification rates change with B and σ in both cases are similar as that in balanced data case. Our way of conformal transformation still works best throughout all the methods. What is different is that this time k_M tends to change in an opposite way along with the density of the support vectors around a specific point. When the ratio of proportions are more extreme, k_M changes in a sharper way. The spatial location of the support vectors in the feature space F is thus taken into consideration.

4.4.2 Ontario Prostate Caner MRI Data

In this section, the proposed method is applied on a prostate cancer MRI data set, continuing as the second phase of study. Still, the study is aiming at finding some statistical methods to classify non-/cancer areas by the imaging data induced by imaging equipments, mainly MR, CT and ultra sound. During the second phase, 9 common

classes are provided and listed in the introduction.

Other situations are similar to those in Chapter 3. No more patients are added into the study. Predictors on each voxels are still the 3 intensity measures from MRI, CT and ultra sound platforms , T2W intensity, ADC intensity and C-Grade intensity. Other measures such as DCE and DWI are only available to part of the patients, and hence are not included in the training process as predictors.

To adopt the proposed data-adaptive scaling in this multi-class case, two-stage SVMs are still required: during the first round, a standard SVM with the selected kernel is conducted, so that the support vectors from the original data set can be found. Based on them, the conformal transformation scalar function can be constructed and the kernel function can be updated. Next, a second-stage SVM is conducted and the resulting estimated boundary will be used as the rule for classification. In terms of choosing appropriate tuning parameters for each method, cross validation is conducted in patient-wise 7-fold for 500 times.

To test the performance, we will compare our proposed method with both traditional and data-adaptive multi-class classification methods. In terms of the traditional methods, one-versus-one (1vs1) and one-versus-all (1vsA) from indirect methods, and the Crammer and Singer's (CS) and He's Simplified SVM (simSVP) direct methods will be included, while for the data-adaptive methods, Amari's and William's adaptively scaling will be included. In terms of the criterion of the classification performance, misclassification rate, percentage of support vectors in the whole data set, F-measure and G-means along with their margins are reported.

The outcome is listed in Table 4.4. Obviously, our proposed method performs almost the best among all the compared methods. A highlight point is that our proposed method has the smallest margins in all performance measures, resulting from the property of the robust decay of the magnification effect from our proposed

data-adaptive kernel. In terms of the accuracy, our proposed method has similar misclassification rate with indirect methods, significantly smaller than the rest methods. F-measure and G-means both have seen the largest values in our proposed method, much larger than other data-adaptive kernel methods. The percentage of support vectors that are used for constructing the classifiers is the smallest in our proposed method as well, much smaller than 1vs1 and 1vsA which have slightly better accuracy than ours and other methods.

It is worth pointing out that among those wrong predicted labels, $G4+4$ is the dominant, in other words, the misclassification always happens in $G4+4$ type cancer. This is because the percentage of this type of cancer is really rare in the training sample, taking only 1-2% among all the labels. This extremely imbalanced data have made it very difficult to detect with a high accuracy. Our proposed method can detect around 60% among this type, while other data adaptive (Amari' s and William's) methods can only find less than 20%, and almost none in other methods.

4.5 Concluding Remarks

In this chapter, we extend our data-dependent SVM construction technique to multi-class situation. Based on the idea of the data-adaptive kernel SVM for binary case, we proposed a new way to construct the data-dependent kernel for the multi-class setting especially when the data are imbalanced, in a way that the decay rates are more robust, and can vary along with the density of the size of neighbours. Thus, the kernel can be adapted optimally for a specific data set. Numerical results have shown the out-performance with our methods.

Table 4.1: *F-measure and G-mean for all five classification methods in Scenario 1 for* $n_1 = 200$, $n_3 = 200$ *and* $n_3 = 200$, *respectively. Maximal margin of error is 0.02.*

		SVM		Wu and Amari (2002)		Williams et al. (2005)		Maratea et al. (2014)		Our method	
B	σ	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean
8	0.1	0.39	0.38	0.43	0.43	0.59	0.59	0.71	0.70	0.78	0.79
8	0.5	0.43	0.42	0.47	0.46	0.66	0.66	0.75	0.75	0.81	0.81
8	5	0.47	0.46	0.53	0.52	0.68	0.68	0.78	0.77	0.84	0.83
40	0.1	0.45	0.45	0.44	0.43	0.61	0.61	0.73	0.72	0.81	0.81
40	0.5	0.53	0.52	0.51	0.50	0.67	0.67	0.75	0.75	0.84	0.83
40	5	0.56	0.55	0.62	0.62	0.71	0.72	0.78	0.78	0.86	0.86
100	0.1	0.52	0.51	0.61	0.59	0.64	0.63	0.78	0.79	0.84	0.85
100	0.5	0.60	0.58	0.67	0.65	0.77	0.67	0.79	0.80	0.86	0.86
100	5	0.69	0.66	0.71	0.70	0.79	0.72	0.81	0.82	0.88	0.88

Table 4.2: *F-measure and G-mean for all five classification methods in Scenario 2 for* $n_1 = 100$, $n_3 = 200$ *and* $n_3 = 300$, *respectively. Maximal margin of error is 0.04.*

		SVM		Wu and Amari (2002)		Williams et al. (2005)		Maratea et al. (2014)		Our method	
B	σ	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean
8	0.1	0.29	0.30	0.40	0.40	0.49	0.49	0.62	0.60	0.77	0.76
8	0.5	0.32	0.32	0.42	0.42	0.58	0.57	0.66	0.65	0.78	0.78
8	5	0.36	0.36	0.48	0.49	0.61	0.60	0.71	0.72	0.80	0.80
40	0.1	0.40	0.40	0.54	0.53	0.57	0.58	0.65	0.66	0.80	0.80
40	0.5	0.50	0.42	0.59	0.58	0.65	0.64	0.68	0.67	0.81	0.80
40	5	0.56	0.56	0.62	0.60	0.68	0.66	0.72	0.73	0.82	0.82
100	0.1	0.42	0.39	0.57	0.55	0.65	0.64	0.66	0.68	0.84	0.83
100	0.5	0.52	0.50	0.63	0.65	0.70	0.69	0.71	0.72	0.85	0.84
100	5	0.59	0.61	0.68	0.70	0.75	0.76	0.75	0.74	0.86	0.87

Table 4.3: *F-measure and G-mean for all five classification methods in Scenario 3 for* $n_1 = 20$, $n_3 = 100$ *and* $n_3 = 480$, *respectively. Maximal margin of error is 0.08.*

		SVM		Wu and Amari (2002)		Williams et al. (2005)		Maratea et al. (2014)		Our method	
B	σ	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean	F-measure	G-mean
8	0.1	0.25	0.23	0.34	0.32	0.45	0.46	0.58	0.57	0.75	0.74
8	0.5	0.28	0.27	0.37	0.38	0.54	0.52	0.62	0.64	0.77	0.77
8	5	0.32	0.29	0.46	0.44	0.57	0.58	0.68	0.66	0.79	0.79
40	0.1	0.35	0.34	0.51	0.49	0.53	0.54	0.60	0.59	0.79	0.78
40	0.5	0.47	0.45	0.55	0.54	0.59	0.58	0.64	0.64	0.80	0.80
40	5	0.51	0.52	0.57	0.55	0.63	0.62	0.68	0.69	0.81	0.81
100	0.1	0.38	0.37	0.54	0.52	0.61	0.60	0.62	0.63	0.82	0.82
100	0.5	0.45	0.45	0.59	0.57	0.65	0.64	0.67	0.67	0.84	0.84
100	5	0.55	0.55	0.62	0.63	0.71	0.71	0.71	0.70	0.85	0.86

Table 4.4: *Outcomes of multi-class prediction on London Cancer Program.*

Methods	Error(%)	SV(%)	F-measure	G-means
Proposed	8.60 ± 0.58	17.46 ± 0.57	0.84 ± 0.05	0.81 ± 0.04
Wu and Amari (2002)	11.88 ± 1.12	21.33 ± 1.27	0.70 ± 0.09	0.66 ± 0.10
Williams et al. (2005)	10.21 ± 0.97	18.93 ± 1.65	0.74 ± 0.12	0.71 ± 0.08
Crammer and Singer (2001)	9.20 ± 1.22	17.57 ± 1.12	0.77 ± 0.06	0.73 ± 0.06
He et al. (2012)	9.33 ± 1.20	18.29 ± 1.07	0.78 ± 0.10	0.74 ± 0.09
1vs1	8.20 ± 1.26	25.41 ± 2.87	0.81 ± 0.06	0.77 ± 0.07
1vsA	8.25 ± 1.57	24.16 ± 2.62	0.82 ± 0.06	0.76 ± 0.06

Chapter 5

Dada-adaptive Kernel-penalized SVM

5.1 Introduction

An support vector machine offers the advantages including lower risk of over-fitting, less model complexity (and hence the improvement of the generalization capability) and less computational cost (Blum and Langley (1997)). The performance of an SVM model relies on selecting the most relevant predictors while removing irrelevant ones when there are many potential predictors. Penalized SVMs offer a way of eliminating redundant predictors, however, the penalized feature selection methods for SVMs in the literature are mostly based on the input space. However, there are possibilities that those predictors which have been eliminated in the input space are useful in the projected feature space, and hence the classifier will lose some useful information.

In this chapter, we propose a novel method of simultaneous feature selection and classification by penalizing data-adaptive kernels in SVMs. Instead of penalizing the standard cost function of an SVM, we will directly penalize an objective function with the data-adaptive kernel function that controls the performance of an SVM. The predictors that are useful in the feature space are selected, and the decision rule can be constructed simultaneously with the predictors. Different penalty terms such

as SCAD, MCP and L_1-norm penalties will be compared. The oracle property of the estimator is proposed. Iterative optimization process will be applied as no analytic form of the estimated coefficients can be obtained. Numerical comparisons show that our model outperforms especially with the imbalanced data.

The rest of the paper organizes as follows. In Section 2, the framework of SVMs and the penalized SVM is introduced. In Section 3, a data-adaptive kernel-penalized SVM is constructed. Not only the oracle property of the coefficients' estimates is proposed, but an algorithm to achieve the goal is introduced for implementation purpose as well. Extensive empirical studies are conducted in Section 4.

5.2 Notation and Framework

Consider a binary classification problem. Given a random sample $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i$ is a vector in the input space $I = R^p$, y_i represents the class index which takes values $+1$ or -1, and p, the dimension of the input space, indicates the number of predictors available in the sample. The goal is to determine a rule such that observations can be labeled into the corresponding class accurately with only limited number of predictors. An SVM can classify the observations by mapping the input data $\mathbf{x}$ into another high-dimensional feature space $F = R^l$, using a nonlinear mapping function $\mathbf{s} : R^p \to R^l$, and searches a linear discriminant function

$$D(\mathbf{x}) \;=\; \mathbf{w}^T \mathbf{s}(\mathbf{x}) + b \tag{5.1}$$

where $\mathbf{w} = (\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_l)$ is an $l-$dimensional vector of parameters, $\mathbf{s}(\mathbf{x}) = (\mathrm{s}_1(\mathbf{x}), \ldots, \mathrm{s}_l(\mathbf{x}))^T$ is the $l-$dimensional column vector, and b is a scalar intercept. $D(\mathbf{x}) = 0$ represents the separating hyperplane in the input space. An individual point $\mathbf{x}$ can be classified by the sign of $D(\mathbf{x})$ as long as the parameters $\mathbf{w}$ and b are determined.

In mathematics, an SVM is the solution to minimizing

$$Q(\mathbf{w}, b, \xi) \;=\; \frac{1}{2}\|\mathbf{w}\|^2 + B\sum_{i=1}^{n}\xi_i \tag{5.2}$$

with respect to $\mathbf{w}$, and b, subject to the constraints

$$y_i(\mathbf{w}^T\mathbf{s}(\mathbf{x}_i) + b) \geq 1 - \xi_i \qquad \text{for } i = 1, \ldots, n,$$

where B is the so-called soft margin parameter that determines the trade-off between the optimal combinatorial choice of the margin and the classification error, and $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)^T$ are nonnegative slack predictors. Equivalently, this optimization problem can be represented in the Lagrangian dual function with the form as

$$\underset{\boldsymbol{\alpha}}{\text{Max}}\; L_D \quad = \quad \underset{\boldsymbol{\alpha}}{\text{Max}}\; \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j < \mathbf{s}(\mathbf{x}_i), \mathbf{s}(\mathbf{x}_j) > . \tag{5.3}$$

subject to the constraints

$$\sum_{i=1}^{n}\alpha_i y_i \;=\; 0,$$

$$0 \;\leq\; \alpha_i \leq\; B$$

for $i = 1, 2, \ldots, n$, where α_i's are the dual predictors (the Lagrange Multipliers) by Lagrange Multiple Methods when solving the minimization problem in (5.2), and $< \cdot\,,\; \cdot >$ is the inner product operator. More details can be found in Chapter 2 and 3. Generally a scalar function $K(\cdot, \cdot)$, which is called a kernel function, is adopted to replace the inner product of the two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ in the dual function in (5.3),

$$\underset{\boldsymbol{\alpha}}{\text{Max}}\; L_D \quad = \quad \underset{\boldsymbol{\alpha}}{\text{Max}}\; \left(\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)\right).$$

Let SV be the set $\{j \mid \alpha_j > 0 \text{ for } j = 1, 2, \ldots, n\}$. Then the corresponding $\mathbf{x}_i$'s where i is in SV are called *support vectors*, where the cardinality of SV, denoted by l, is the dimension of the feature space F. As pointed out in Chapter 2 and 3, α_i's are representing the contribution of the corresponding support vectors, and the kernel form of SVM can be written as

$$D(\mathbf{x}) \;=\; \sum_{i \in SV}\alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

The estimated intercept b_j obtained by using the jth support vector $\mathbf{x}_j$ is defined as

$$b_j \;=\; y_j - \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j).$$

Hastie et al. (2001) have proved that for different j in the support vectors set SV, the b_j is the same. In practice, we can take the average of all the b_j's as the estimate of the intercept b. Quite a few typical kernels are available, such as the radial form

$$K(\mathbf{x}, \mathbf{z}) = f(-\|\mathbf{x} - \mathbf{z}\|^2),$$

among which the most popular one is the Gaussian RBF kernel with the bandwidth parameter σ

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|^2 / 2\sigma^2). \tag{5.4}$$

5.2.1 Data-Adaptive Kernel SVM

In Chapter 3, we have proposed the data-adaptive kernel SVM to increase the separability between two categories, and the spatial resolution around the boundary surface is enhanced and so is the separability. This is especially important when the data are imbalanced, since we have demonstrated that the imbalance in the data can severely affect the performance of an SVM in Chapter 3. Geometrically speaking, when the input space I is the Euclidean space, the Riemannian metric is induced in the feature space F by the mapping $\mathbf{s}$. In Chapter 3, Result 3.3.1 demonstrates the connection between a kernel function K and a mapping $\mathbf{s}$, and a data-adaptive kernel function is constructed to enhance the accuracy of an SVM based on the Gaussian RBF kernel function. Let $C(\mathbf{x}, \mathbf{x}')$ be a positive scalar function such that

$$C(\mathbf{x}, \mathbf{x}') \;=\; c(\mathbf{x})c(\mathbf{x}'),$$

where $\mathbf{x}$ and $\mathbf{x}'$ are vectors from the input space, and $c(\mathbf{x})$ is a positive univariate scalar function. The kernel function K is updated as

$$\tilde{K}(\mathbf{x}, \mathbf{x}') \;=\; C(\mathbf{x}, \mathbf{x}')K(\mathbf{x}, \mathbf{x}') \;=\; c(\mathbf{x})K(\mathbf{x}, \mathbf{x}')c(\mathbf{x}'), \tag{5.5}$$

where $\tilde{K}(\mathbf{x}, \mathbf{x}')$ corresponds to the mapping $\tilde{\mathbf{s}}$ that satisfies the transformation

$$\tilde{\mathrm{s}}_{ij}(\mathbf{x}) \;=\; c_{ij}(\mathbf{x})\mathrm{s}_{ij}(\mathbf{x}),$$

where $c_{ij}(\mathbf{x}) = \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}C(\mathbf{x}, \mathbf{z})|_{\mathbf{z}=\mathbf{x}}$. The above process is known as the *adaptive scaling*. In Chapter 3, we propose to adaptively scale the primary kernel function K by constructing $c(\mathbf{x})$ with the L_1-norm radial basis function

$$c(\mathbf{x}) \;=\; e^{-|D(\mathbf{x})|\cdot k_M(\mathbf{x})} \tag{5.6}$$

and

$$k_M(\mathbf{x}) \;=\; AVG_{i\in\{j:\|\mathbf{s}(\mathbf{x}_j)-\mathbf{s}(\mathbf{x})\|^2<M,\ y_j\neq y\}}(\|\mathbf{s}(\mathbf{x}_i)-\mathbf{s}(\mathbf{x})\|^2),$$

where $D(\mathbf{x})$ is given by (5.1), AVG denotes the average operator, y is the class label associated with $\mathbf{x}$, and M can be regarded as the distance between the nearest and the farthest support vectors from $\mathbf{s}(\mathbf{x})$. Theorem 3.3.2 and 3.3.3 give numerical results of the updated mapping $\tilde{\mathrm{s}}$ by the data-adaptive kernel function, and the magnification effect of the spatial resolution by the updated mapping $\tilde{\mathrm{s}}$ can be calculated accordingly. The magnification effect is roughly the largest near the separating boundary, and it decreases robustly with a slow and steady rate from near the separating boundary to faraway locations. This process is important when the data are imbalanced, since by incorporating $k_M(\mathbf{x})$ into $c(\mathbf{x})$, the adaptive scaling process updates the spatial information, and is proved to have greater separability even when the data are imbalanced in Chapter 3 and 4.

5.2.2 Penalized SVM

Another issue that needs to be considered during the classification process is the complexity of the model. More straightforwardly, the number of the predictors that are used to construct the classifier needs to be limited. When redundant predictors are involved, extra noisy information will be introduced and hence deteriorate the accuracy of the classifier (Zhang et al. (2016)). Fortunately, the number of the predictors

can be controlled in the SVM framework. Under the standard prediction risk framework of loss plus penalty form (Hastie et al. (2001)), the potential misclassification cost can be specified by a universal weight q for each of the sample point from the two classes, namely, $Q_i = q$ if $y_i = 1$ and $Q_i = 1 - q$ if $y_i = -1$ for some $0 < q < 1$. The classification boundary can be estimated by a linear weighted SVM (Lin (2002)), by solving

$$\min_{\mathbf{w},\ \mathrm{w}_0} Loss(\mathbf{w}) \quad = \quad \min_{\mathbf{w},\ \mathrm{w}_0} n^{-1} \sum_{i=1}^{n} Q_i (1 - y_i(\mathbf{x}_i^T \mathbf{w} + \mathrm{w}_0))_+ \ + \ \lambda \mathbf{w}^T \mathbf{w},$$

where $(1-t)_+ = \max\{1-t, 0\}$ denotes the hinge loss, $\mathbf{w}$ are the coefficients of the predictors, w_0 is the intercept and λ is a positive regularization parameter. When the weight $q = 0.5$, the linear weighted SVM reduces to the standard SVM (Lin (2002)). With the hinge loss of the form $E(Q(1 - (y\mathbf{X}^T\mathbf{w} + \mathrm{w}_0))_+$, a clear analytic form of estimators of the coefficients in the decision boundary 5.1 is given in the following

$$\hat{\mathbf{w}} = \arg \min_{\mathrm{W}_0, \mathbf{w}} \quad n^{-1} \sum_{i=1}^{n} Q_i (1 - y_i(\mathbf{Z}_i^T \mathbf{w} + \mathrm{w}_0))_+.$$

Furthermore, when selecting predictors from the input space, it is often assumed that the true model has sparse predictors, or equivalently, $\mathbf{w}^T = (\mathbf{w}_{true}^T, \mathbf{0}^T)$, where $\mathbf{w}_{true}^T = (\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_h)$. Denote $\mathbf{x}_i^T = (\mathbf{z}_i^T, \mathbf{u}_i^T)$, where the $h \times 1$ vector $\mathbf{z}_i$ is the nonzero-coefficient part and the $(p-h) \times 1$ vector $\mathbf{u}_i$ corresponds to the redundant information. To obtain the sparse estimator of the coefficients, Zhang et al. (2016) proposed that the penalty terms should be added directly to the loss function in the way that

$$Loss(\mathbf{w}) = n^{-1} \sum_{i=1}^{n} Q_i (1 - y_i(\mathbf{x}_i^T \mathbf{w} + \mathrm{w}_0))_+ \ + \ \sum_{j=1}^{p} p_{\lambda_n}(\|\mathrm{w}_j\|), \tag{5.7}$$

where $p_{\lambda_n}(\cdot)$ is a symmetric, non-convex penalizing function with some tuning parameter λ_n. The oracle properties of the estimators obtained by minimizing (5.7) were proved under some regulatory conditions. Some popular penalty functions were taken as examples in their study, such as the smoothly clipped absolute deviance (SCAD) penalty (Fan and Li (2001)) and the minimax concave penalty (MCP, Zhang (2010)).

However, such a feature selection process cannot guarantee the classification accuracy of the SVM. As has pointed in Lemma 3.3.1, it is the kernel function that actually controls the classifier's performance. When the original input space is projected into the feature space, the kernel presents a critical part, and those predictors with very little or no information in the feature space F should be eliminated. Thus, a straightforward idea to select predictors during the training of the SVM is to directly penalize the objective function that contains the kernel. Thus, we propose a new method of simultaneous classification and feature selection process by penalizing data-adaptive kernels in SVM, which is called the *data-adaptive kernel-penalized SVM*.

5.3 Methodology

In this section, a data-adaptive kernel-penalized SVM is proposed. The method can simultaneously select predictors and conduct classification with an data-adaptive kernel function. Instead of adding penalty to the standard hinge loss function, we propose to add the penalty term directly to the SVM of the kernel formulation, so that the number of the predictors that are truly useful in the feature space can be controlled. To put the issue of imbalance data into consideration, the data-adaptive kernel will be used. The oracle properties of the estimates of the true parameters under our proposed setting is proved.

5.3.1 Kernel-Based Parameters

We first introduce the parameters derived from the kernel function in our methodology. In the following sections, we focus on the Gaussian RBF kernel as (5.4), where the parameter σ is universal for all components of the input vectors $\mathbf{x}$ and $\mathbf{z}$. When there are more than one predictors available in the input space, each component of the parameter vector $\boldsymbol{\sigma}$ in the Gaussian RBF kernel can take different values for different

predictors

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\sum_{j=1}^{p}(x_j - z_j)^2/2\sigma_j^2),$$

where p is the dimension of the input space I (Maldonado and Weber (2009)). Consequently, the contributions of the corresponding predictors can be determined by the parameters $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_p)$. For instance, if σ_j is very large, the $j-$th predictor tends to contribute very little to the kernel function as the corresponding component in the exponent will be close to zero. Contrarily, if σ_j is small, the contribution of the $j-$th predictor will be large and its importance increases consequently. Thus, by controlling the $j-$th component in the parameter vector $\boldsymbol{\sigma}$, the importance of the $j-$th predictor can be determined. This provides a way of selecting predictors by directly estimating the parameters in the kernel function. Accordingly, the following change of the kernel function is proposed as

$$K(\mathbf{x}, \mathbf{z}; \mathbf{w}) = \exp\{-\mathbf{w} \otimes \|(\mathbf{x} - \mathbf{z})\|^2\}, \tag{5.8}$$

where $\mathbf{w} = (\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_p) = 1/\boldsymbol{\sigma}^2 = (1/\sigma_1, 1/\sigma_2, \ldots, 1/\sigma_p)^2$ and $\otimes$ represents the component-wise product. When w_j is large, the contribution of the $j-$th predictor will be large and hence its importance increases. Contrarily, when w_j is very small, the $j-$th predictor tends to contribute very little to the kernel function, and should not be included during the training of an SVM. However, even if the absolute value of w_j is really small but is not zero, its influence in the kernel function still exists. Including too many active predictors in the classifier will dramatically complicate the model, which may result in extra noisy information. The best way may be to force some of these predictors to be exactly zero. This can be achieved by adding the penalty item, and the number of active predictors can be sparse.

5.3.2 Data-adaptive Kernel-penalized SVM

To control the number of predictors in the classifier, the penalty term for each component of the parameters $p_{\lambda_n}(|\mathrm{w}_j|),\ j = 1, 2, \ldots, p$, will be included. Since the performance of an SVM depends on the kernel function, we propose to add the penalty

terms directly to the dual maximization problem that contains the kernel function. Accordingly, the data-adaptive kernel-penalized SVM is initially proposed as the solution to

$$\underset{\boldsymbol{\alpha},\ \mathbf{w}}{\text{Max}}\ L_D \ = \ \left(\underset{\boldsymbol{\alpha},\ \mathbf{w}}{\text{Max}} \ \ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \tilde{K}(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}) - \sum_{j=1}^{p} p_{\lambda_n}(|\mathrm{w}_j|) \right), \tag{5.9}$$

such that

$$\begin{aligned} \sum_{i=1}^{l} \alpha_i y_i \ &= \ 0, \\ 0 \ \leq \ \alpha_i \ &\leq B, \qquad i = 1, 2, \ldots, l \\ \mathrm{w}_j \ &\geq \ 0, \qquad j = 1, 2, \ldots, p, \end{aligned}$$

where $\tilde{K}(\mathbf{x}, \mathbf{z})$ is the data-adaptive kernel function from (5.5), $c(\mathbf{x})$ in $\tilde{K}(\mathbf{x}, \mathbf{z})$ is from (5.6) and the primary kernel function is from (5.8). When the estimate of $\widehat{\mathbf{w}}$ is obtained, the predictors with non-zero coefficients are considered to be the truly active predictors that will affect the decision boundary. The boundary will be estimated by

$$\widehat{D}(\mathbf{x}) \ = \ \sum_{i \in SV} \alpha_i y_i \tilde{K}(\mathbf{x}_i, \mathbf{x}_j; \widehat{\mathbf{w}}) + \widehat{b} \tag{5.10}$$

and the intercept b can be replaced by using any support vector $\mathbf{x}_j$ as

$$\widehat{b} \ = \ y_j - \sum_{i \in SV} \alpha_i y_i \tilde{K}(\mathbf{x}_i, \mathbf{x}_j; \widehat{\mathbf{w}}).$$

With the decision rule in (5.10), a test observation $\mathbf{x}$ can be assigned to the class by the sign of $\widehat{D}(\mathbf{x})$.

There are several options for the specific forms of the penalties. In general, we consider the non-convex penalty which satisfies Assumptions 1 and 2 in the appendix. Such a non-convex penalty term is motivated by the fact that the L_1-penalty, generally known as the least absolute shrinkage and selection operator, or $LASSO$, does not have the oracle property due to the over-penalization on large coefficients, and hence L_1-penalty is not a proper choice when the relevant predictors are to be selected among the space with higher dimensions in classification (Zhang et al. (2016)).

Note that the tuning parameter λ can depend on the sample size n. Several popular non-convex penalty terms satisfying Assumptions 1 and 2 are:

1. SCAD: Smoothly Clipped Absolute Deviation (Fan and Li (2001))

$$\begin{aligned} p_\lambda(|\mathbf{w}|) &= \lambda|\mathbf{w}|I(0 \le |\mathbf{w}| < \lambda) \\ &+ \frac{a\lambda|\mathbf{w}| - (\mathbf{w}^2+\lambda^2)/2}{a-1} I(\lambda \le |\mathbf{w}| \le a\lambda) + \frac{(a+1)\lambda^2}{2} I(|\mathbf{w}| > a\lambda) \end{aligned}$$

   for some $a > 2$.

2. MCP: Mini-max Concave Penalty (Zhang (2010))

$$p_\lambda(|\mathbf{w}|) = \lambda(|\mathbf{w}| - \frac{\mathbf{w}^2}{2a\lambda})I(0 \le |\mathbf{w}| < a\lambda) + \frac{a\lambda^2}{2} I(|\mathbf{w}| \ge a\lambda) \qquad \text{for some a} > 1.$$

3. L_0-norm smooth approximation: $\|\mathbf{w}\|_0 = |\{i : \mathrm{w}_i \neq 0\}|$ by (Maldonado et al. (2011)). Unlike L_p-norm with $p > 0$, L_0-norm is not precisely a norm because the triangle inequality does not hold and consequently it is not smooth. Thus the approximation by a concave function is applied on the L_0-norm so that a penalty function is

$$p_\lambda(|\mathbf{w}|) = \mathbf{1}^T(\mathbf{1} - \exp(\lambda|\mathbf{w}|)) \approx \|\mathbf{w}\|_0,$$

   where λ is an approximation parameter.

Remarks: 1. Penalty terms are directly added to the loss function in literature. However, the standard loss function does not contain the kernel function. When the data are imbalanced, the performance of a standard SVM will be affected. Consequently, the predictors selected without consideration of the imbalanced data may be unreliable. Contrarily, the data-adaptive kernel-penalized SVM can fulfill the feature selection process while taking the imbalance of data into account. 2. The specific form of the kernel function is not limited to the radial kernels. Other types of kernels such as the polynomial kernel $K(\mathbf{x}, \mathbf{z}) = (1 + \sum_{j=1}^{p} x_j z_j)^d$ are also available to describe the mapping by kernels. However, not all the kernels are feasible for simultaneous classification and feature selection process because of technical difficulty. For

example, polynomial kernels are determined only by the order parameter d, while it is not obvious how feature selection can be conducted during the classification process. However, our method is still very attractive in applications, since the Gaussian RBF kernel we adopted in our method may be the most popular kernel. 3. The constraints in the dual function contain the nonnegativity of the parameters $\mathbf{w}$, because originally in the Gaussian kernels, the parameter $\boldsymbol{\sigma}$ is naturally considered as a nonnegative standard deviation. However, we should notice that, in our setting, we can remove nonnegativity by simply using a quadratic form of the parameters in the penalized kernels so that whether the parameters are positive or negative will not affect the decision boundary.

5.3.3 An Algorithm to Solve Data-Adaptive Kernel-penalized SVM

To solve the data-adaptive kernel-penalized SVM in (5.9), a two-stage algorithm is proposed. The whole process is similar to the algorithm for data-adaptive kernel SVM in Chapter 3. In the first stage, a standard SVM is solved so that the location information of the support vectors can be found. Based on the location information, the primary kernel function is updated adaptively by (5.5). In the second stage, a optimization problem with the updated kernel and the penalty item is solved.

Since no analytic form of the estimates can be found, an iterative procedure is adopted to solve the optimization problem (Maldonado et al. (2011)). To be specific, in the $t-$th round iteration, $t = 1, 2, \ldots, T$, a standard dual optimization problem for an SVM with the $(t-1)-$th estimated kernel parameter vector $\widehat{\mathbf{w}}^{(t-1)}$, is to be solved as

$$\underset{\boldsymbol{\alpha}}{\text{Max}}\ L_1(\boldsymbol{\alpha}) \quad = \quad \underset{\boldsymbol{\alpha}}{\text{Max}}\ \left(\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j; \widehat{\mathbf{w}}^{(t-1)}) \right) \tag{5.11}$$

such that

$$\begin{aligned} \sum_{i=1}^{n} \alpha_i y_i &= 0, \\ 0 \leq \alpha_i &\leq B, \qquad i = 1, 2, \ldots, n, \end{aligned}$$

and the result is denoted as $\boldsymbol{\alpha}^{(t)}$. During this stage, the support vectors are obtained by those non-zero α_i's, and the dimension of the feature space $l^{(t)}$, which is the cardinality of the set of active α_i's in the $t-$th iteration, is found. Correspondingly, $c(\mathbf{x})$ can be constructed as (5.6) so that the data-adaptive kernel function $K(\mathbf{x}, \mathbf{z})$ can be updated as $\tilde{K}(\mathbf{x}, \mathbf{z})$ by (5.5). In the second stage of the $t-$th iteration, a non-linear formulation with the fixed $\boldsymbol{\alpha}^{(t)}$ is solved in the following

$$\underset{\mathbf{w}}{\text{Min}}\ L_2(\mathbf{w}) \quad = \quad \underset{\mathbf{w}}{\text{Min}} \quad \left(\frac{1}{2} \sum_{i=1}^{l^{(t)}} \sum_{j=1}^{l^{(t)}} \alpha_i^{(t)} \alpha_j^{(t)} y_i y_j \tilde{K}(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}) + \sum_{j=1}^{p} p_{\lambda_n}(|\mathrm{w}_j|) \right) \quad (5.12)$$

such that

$$\mathrm{w}_j \geq 0, \qquad j = 1, 2, \ldots, p.$$

where the result is denoted as $\widehat{\mathbf{w}}^{(t)}$. The optimization in the second stage aims at eliminating as many components in $\mathbf{w}$ as possible so that the solution can be sparse. The whole process will stop when $\|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\|$ is sufficiently small. The penalty function can be arbitrarily any form as long as Assumptions 1 and 2 in the appendix are satisfied.

5.3.4 The Oracle Property

In this subsection, we will study the oracle property of the estimator. We will show that, given some regularity conditions, the distance between the estimates and the true values of the parameters goes to 0 when the sample size is sufficient large. Here we only need to consider the optimization process in the second stage in (5.12), since all the unknown information regarding the parameters $\mathbf{w}$ is included in this stage (Note that $\boldsymbol{\alpha}$ is considered as a fixed constant vector in the second stage). Define the

estimator

$$\hat{\mathbf{w}} = \arg\min\ L_2(\mathbf{w}) \ = \ \arg\min\ \left(\sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}) + \sum_{j=1}^{p} p_{\lambda_n}(|\mathrm{w}_j|) \right).$$

Then under some regularity conditions in the appendix, the local oracle property is proposed.

Theorem 5.3.1. *Assume that Conditions 1-5 and Assumptions 1-2 for penalty terms are all satisfied. If* $\max\{|p''_{\lambda_n}(w_j)| : w_j \neq 0\} \to 0$, *then there exists a local minimizer* $\hat{\boldsymbol{w}}$ *of* $L_2(\boldsymbol{w})$ *such that* $\|\hat{\boldsymbol{w}} - \boldsymbol{w}_{true}\| = O_p\{\sqrt{q/n}\}$, *where* $\boldsymbol{w}_{true}$ *is the true parameters of the predictors.*

The detailed proof is provided in the appendix. Theorem 5.3.1 guarantees that the estimate of the parameter in our proposed method acts as if the true values of the parameters were known. When the sample size is sufficiently large, the distance between the estimates and the true values of the parameters is so small that it can be ignored. Consequently, the estimated decision rule in (5.10) can be obtained as if the true one were already known, and we can use it to classify new observations.

Though various approaches for SVM-based feature selection procedures are available, our proposed method is different from most of what currently exists in literature. As previously introduced in Chapter 2, the wrapper methods have a different methodological motivation of selecting predictors. The wrapper methods find a subset of predictors by ranking them according to some criteria until some specific stopping rule is met, and then construct the decision boundary on the basis of selected predictors and classify new objects. This process separates the processes of classification and feature selection, and the ranking system is difficult to unify for different scenarios. Our proposed method can directly obtain an minimal subset of predictors and simultaneously classify objects, by penalizing the kernel function and eliminating noisy predictors, without ranking the importance of the predictors. The process of the proposed method is more time-efficient compared to the wrapper methods, and the method improves the classification performance especially when the data are im-

balanced.

Our proposed method is different from those methods that penalize the weight vectors of SVMs or the original loss objective function, which shows little relation to the kernel function. These methods apply only for linear or polynomial kernels, which limits the use of the SVM and its capability of generalization. Instead of putting penalty terms on the standard loss function, our method directly puts the penalized term on the Lagrangian dual function which contains the explicit form of a kernel function. This enables us to apply the data-adaptive scaling process on the primary kernel functions. Hence, even if the data are imbalanced, the performance of classification with the SVM is still excellent. This enhances the robustness and reliability of the classification process with an SVM.

5.4 Experiment Results

In this section, a set of simulation studies are carried out to assess the accuracy of the data-adaptive kernel-penalized SVM. We will compare the performance of our proposed data-adaptive kernel-penalized SVMs to the performance of other penalized SVMs with penalties directly on the loss function. For our data-adaptive kernel-penalized SVM, we use penalties of SCAD (DA-SCAD-SVM) and MCP (DA-MCP-SVM). For other penalized SVM, we use penalties of SCAD (SCAD-SVM, Zhang et al. (2016)), MCP (MCP-SVM, Zhang et al. (2016)), L_1-norm (L_1-SVM, Zhu et al. (2003)), adaptively weighted L_1-norm with a weight parameter $w = 0.5$ (Adapt L_1-SVM, Zou (2007)) and L_0-norm approximation (L_0-SVM, Maldonado et al. (2011)). The comparisons are made under various levels of imbalance in data. The main target is to test the ability of identifying the relevant predictors and controlling the test error when the data are both balanced and imbalanced.

5.4.1 Simulation Study

We consider the data generation process of a standard discriminant analysis, which follows the setting from Park et al. (2012) and Zhang et al. (2016). The model is described as $Pr(Y = 1) = w$ while $Pr(Y = -1) = 1 - w$, where w will control the imbalance level. The input predictors $X|(Y = 1) \sim MVN(\boldsymbol{\mu}, \Sigma)$ and $X|(Y = -1) \sim MVN(-\boldsymbol{\mu}, \Sigma)$, $\boldsymbol{\mu} = (0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0, \ldots,\ 0)^T \in \mathbf{R}^p$, $\Sigma = (\sigma_{ij})$ with diagonal elements $\sigma_{ii} = 1$ for $i = 1, 2, \ldots, p$ and $\sigma_{ij} = \rho = -0.2$ for $1 \leq i \neq j \leq q$, and q is set as 5. The label is determined by $sgn(1.5X_1 + 2.3X_2 + 2.8X_3 + 3.3X_4 + 3.8X_5)$.

In terms of tuning regularization parameters for any approach mentioned above, we adopt the procedure similar to Mazumder et al. (2011). The prediction error is calculated by 5-fold method. That is, 4/5 of all the sample points will be randomly selected and used as the training set, while the left 1/5 of the sample points will be used as the test set to calculate the prediction error. An initial guess of $\mathbf{w}$ is set as $\mathbf{1}^T$. During the second stage of solving the data-adaptive kernel-penalized SVM, the gradient descent procedure is adopted for this non-linear optimization problem. The iterative algorithm will stop if the change in the estimates of the predictors $\mathbf{w}$ in two consecutive rounds, namely $\|\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}\|$, is smaller than a given threshold ϵ.

For the tuning parameter λ in the penalty term, we use the SVM-extended Bayesian information criterion (SVMIC) proposed in Zhang et al. (2016) as

$$SVMIC_\gamma(S) = \sum_{i=1}^{n} 2w\xi_i + \log n|S| + 2\gamma\binom{p}{|S|}, \tag{5.13}$$

where $\xi_i,\ i = 1, 2, \ldots, n$, are the optimal slack predictors correspondingly, S is a subset of $\{1, 2, \ldots, p\}$, $|S|$ is the cardinality or the size of S, and $\binom{\cdot}{\cdot}$ represents the combination operator. The idea is motivated by the standard Bayesian information criterion and is extended by Chen and Chen (2008). The range of λ is set as $\{2^{-6}, 2^{-5}, \ldots, 2^3\}$, and γ is set as 0.5 in the tuning procedure without loss of generality (Chen and Chen (2008)). The value of λ will be set as the one that maximizes (5.13). Note that the values of the slack variables ξ_i in (5.13) are not available directly, but they can be

calculated by $\xi_i = [1 - y_i \widehat{D}(\mathbf{x}_i)]_+$ for $i = 1, \ldots, n$, where $[t]_+ = \max\{0, t\}$, and $\widehat{D}(\mathbf{x}_i)$ can be obtained by (5.10) (Claeskens et al. (2008)). Due to computational consideration, the cost parameter B takes values 0.1, 0.5, 1, 5, 10, 20, 50, 80, 100, 200, 500. The bandwidth parameter M in the data-adaptive kernel function is determined by minimizing the test error in the 5-fold cross validation process.

Further, we use w to represent the proportion of the class labeled as '1' in different scenarios, where w shows the imbalance level. As suggested in Zhang et al. (2016), for $SCAD$ and MCP penalties, the constant a will be set as 3.7 and 3, respectively.

Table 5.1, 5.2, 5.3 and 5.4 summarize the performances with different combinations of imbalance level and the number of predictors, based on a replication of 100 times. The sample sizes n are fixed as 100 and 400, respectively. Columns of 'Relevant' and 'Irrelevant' show the information of the mean values of the truly active and inactive predictors selected by the model, respectively. Column 'True' gives the percentage that the true model containing exactly those 5 active predictors is correctly selected during the 100 replications. Values in parentheses are the corresponding empirical standard errors.

In general, the SVMs with the penalized data-adaptive kernels show a much greater probability of correctly selecting the true model as n increases, which is consistent with the asymptotic oracle property. According to the numbers in Column *Relevant*, the SVMs with penalties of $SCAD$ and MCP find the most relevant predictors compared with other methods. The SVM with L_0-norm approximation can find some relevant predictors, while the SVMs with L_1-norm penalty tend to fail in selecting the correct predictors, with or without adaptive weights. According to Column *Irrelevant*, the two data-adaptive kernel-penalized methods exclude most irrelevant predictors and hence eliminate the noisy predictors. The missing relevant predictor, if there is any, is mostly from X_1 due to the setting that X_1 has the weakest effect.

On the other hand, when the imbalance level of data is increasing, the prediction error tends to increase, which is consistent with the findings in Chapter 3. However, given a specific level of imbalance in data, test prediction errors from data-adaptive kernel-penalized SVMs are universally smaller than those obtained from other approaches, because these two methods give the fewest noisy predictors so that the prediction error is minimized. More importantly, when the imbalance level increases, our data-adaptive kernel-penalized SVMs outperform among all methods, which agrees with the finding in Chapter 3 that the data-adaptive kernel can improve the classification performance. This adaptive scaling process on the kernel is only applicable to our setting and not to any other method due to lack of kernel functions in the model structures (penalized SVMs have penalty terms directly on the loss function, which is not described in the kernel form). At the mean time, the feature selection performance tends to be changed little, especially in the non-convex penalized data-adaptive kernel SVMs.

It is worth noting that the combination (n, p) shows that, even when the number of potential predictors is proportional to the sample size or larger, our methods still performs well. This gives us some clue that the method may still work in big data or ultrahigh dimensional settings. Indeed, the oracle property in our proposed method indicates that, the true predictors can still be selected even when the dimension of the input space is larger than the sample size, which is exactly the ultra-dimensional setting.

5.4.2 A Real Data Example

In this study, we use an open-to-public Wisconsin Breast Cancer (WBC) data set from the UCI repository (Blake and Merz (1998)), which contains 569 observations (212 malignant and 357 benign tumors) that are described by 30 continuous predictors. These predictors are measured by a digitized image of a Fine Needle Aspirate

(FNA) of a breast mass, which can describe predictors of the cell nuclei shown in the images. As a pre-process step, the predictors were first standardized.

Different methods will be compared, both with and without penalties. For classifiers without penalties, the Gaussian kernel will be adopted, and all of the input predictors will be used to estimate the decision boundary. For those with penalties, we will use data-adaptive kernel-penalized SVMs with SCAD and MCP penalties, as well as the penalized SVMs with SCAD and MCP penalties, which gave the best performances in the simulation study. The number of predictors selected and test errors will be reported. For those approaches that require 2-stage optimization process, the solutions for the 1st stage optimization process are used as the initial values for the 2nd stage optimization if needed (such as our proposed method). For SCAD and MCP penalties, the constant a is still fixed as 3.7 and 3 respectively, the same as the values used in the simulation process. A $5-$fold cross validation will be conducted to obtain the budget parameter B, the local bandwidth parameter M and the penalty parameter λ.

Table 5.5 summarizes the classification outcome of the mean and the standard deviation (in parentheses) of the prediction error and the number of predictors selected with different approaches. It is clear that the data-adaptive kernel-penalized SVMs perform the best among all approaches, with a significantly lower prediction error and number of predictors selected than any other method. Compared with penalized SVM with SCAD and MCP penalties, data-adaptive kernel-penalized SVMs with the corresponding penalties still outperform, even though the penalties are the same. MCP seems to be a better choice for the penalty term, since the number of the predictor is the smallest, and the standard deviation is smaller. Adaptively weighted L_1-norm SVM and L_1-norm SVM are fair. Clearly, the numerical results have confirmed that data-adaptive kernel-penalized SVMs with SCAD or MCP penalty are both promising classifiers with low prediction error and excellent feature selection ability.

5.5 Concluding Remarks

In this chapter, we propose the data-adaptive kernel-penalized SVM, a new method that simultaneously achieves classification and feature selection, especially when the data are imbalanced. Instead of penalizing the loss function of SVMs, an non-convex penalty is proposed to be added directly to the SVM formulation with the kernel function. The benefit is that the truly active predictors are more likely to be selected in the feature space instead of the input space, because it is the kernel function that mainly determines the classification process. The data-adaptive kernel is adopted to the SVM so that even when the data are imbalanced, the performance of the SVM is still excellent. Along with the oracle properties as if the true sparsity in the feature space is already known, our proposed method works well in both simulation study and the real data example, possibly even when the ultra-dimensional setting exists.

The method proposed in this chapter is actually an embedded approach, as mentioned in the introduction part, and the forms of penalty terms are not limited to those applied in the methodology above. The methodology may be extended to the multi-category classification problem, though the data-adaptive kernels need to be modified. Another issue is the choice of the primary kernel function. The methodology proposed is based on the Gaussian RBF kernel because of its natural link with the contribution of the predictors. Extensions will be considered in the future work.

Table 5.1: *Simulation outcomes for data-adaptive kernel-penalized SVMs and penalized SVMs with sample size* $n = 100$. *Margins are in brackets.*

Method	Proportion	p	Relevant	Irrelevant	True%	Test Error%
DA-SCAD-SVM	w=0.50	50	5.00(0.00)	0.88(0.16)	96	8.16(0.20)
		100	5.00(0.00)	0.91(0.14)	96	8.72(0.20)
	w=0.75	50	4.96(0.01)	0.92(0.23)	94	9.23(0.30)
		100	4.95(0.01)	0.95(0.27)	94	9.85(0.30)
	w=0.90	100	4.91(0.03)	1.10(0.39)	91	10.55(0.40)
		100	4.90(0.03)	1.09(0.41)	91	10.93(0.40)
DA-MCP-SVM	w=0.50	50	5.00(0.00)	0.12(0.01)	98	7.20(0.20)
		100	5.00(0.00)	0.13(0.01)	98	7.38(0.20)
	w=0.75	50	4.98(0.01)	0.26(0.03)	96	8.44(0.20)
		100	4.98(0.01)	0.28(0.03)	96	8.90(0.20)
	w=0.90	100	4.95(0.02)	0.42(0.04)	92	9.20(0.30)
		100	4.94(0.02)	0.45(0.04)	92	9.65(0.30)
SCAD-SVM	w=0.50	50	4.92(0.02)	1.92(0.18)	96	8.23(0.20)
		100	4.91(0.02)	1.99(0.17)	96	8.66(0.20)
	w=0.75	50	4.83(0.03)	2.01(0.31)	91	10.19(0.40)
		100	4.78(0.04)	2.13(0.36)	91	10.87(0.40)
	w=0.90	100	4.76(0.04)	3.35(0.41)	88	12.15(0.50)
		100	4.74(0.04)	3.40(0.43)	87	12.36(0.50)
MCP-SVM	w=0.50	50	5.00(0.00)	0.27(0.02)	98	7.32(0.20)
		100	5.00(0.00)	0.29(0.02)	98	7.41(0.20)
	w=0.75	50	4.92(0.01)	0.43(0.03)	93	8.96(0.20)
		100	4.91(0.01)	0.47(0.03)	93	9.29(0.30)
	w=0.90	100	4.85(0.03)	0.88(0.05)	89	10.63(0.40)
		100	4.84(0.03)	0.91(0.05)	89	11.79(0.40)

Table 5.2: *Simulation outcomes for non-penalized SVMs with sample size* $n = 100$. *Margins are in brackets.*

Method	Proportion	p	Relevant	Irrelevant	True%	Test Error%
Adapt L_1-SVM	w=0.50	50	4.85(0.02)	2.87(0.66)	62	12.16(0.50)
		100	4.78(0.04)	2.93(0.49)	54	14.16(0.40)
	w=0.75	50	4.61(0.04)	4.11(0.23)	55	13.88(0.40)
		100	4.37(0.08)	4.23(0.56)	43	15.73(0.40)
	w=0.90	50	4.33(0.07)	6.28(0.77)	41	16.68(0.50)
		100	4.03(0.10)	6.79(0.78)	25	17.02(0.50)
L_1-SVM	w=0.50	50	4.38(0.07)	13.62(0.90)	23	16.28(0.50)
		100	4.01(0.10)	13.10(0.86)	5	20.23(0.50)
	w=0.75	50	4.13(0.09)	15.18(1.05)	8	18.71(0.50)
		100	3.91(0.10)	14.92(1.03)	0	22.33(0.60)
	w=0.90	50	3.87(0.10)	16.99(1.22)	2	20.02(0.60)
		100	3.81(0.13)	16.87(1.21)	0	25.01(0.70)
L_0-SVM	w=0.50	50	4.86(0.05)	31.08(1.52)	10	16.67(0.50)
		100	4.71(0.06)	42.98(2.13)	4	19.33(0.60)
	w=0.75	50	4.62(0.07)	35.71(1.67)	3	19.18(0.60)
		100	4.45(0.08)	46.29(2.20)	0	22.00(0.80)
	w=0.90	50	4.33(0.10)	39.53(2.02)	1	22.61(0.80)
		100	4.02(0.10)	59.01(2.54)	0	25.98(1.00)

Table 5.3: *Simulation outcomes for data-adaptive kernel-penalized SVMs and penalized SVMs with sample size $n = 400$. Margins are in brackets.*

Method	Proportion	p	Relevant	Irrelevant	True%	Test Error%
DA-SCAD-SVM	w=0.50	200	5.00(0.00)	0.58(0.11)	98	7.76(0.20)
		400	5.00(0.00)	0.72(0.13)	98	8.13(0.20)
	w=0.75	200	4.98(0.01)	0.67(0.12)	96	8.76(0.30)
		400	4.98(0.01)	0.71(0.13)	96	9.12(0.30)
	w=0.90	200	4.95(0.02)	0.81(0.17)	93	9.14(0.30)
		400	4.94(0.02)	0.77(0.16)	93	9.93(0.30)
DA-MCP-SVM	w=0.50	200	5.00(0.00)	0.05(0.01)	98	6.28(0.20)
		400	5.00(0.00)	0.06(0.01)	98	6.91(0.20)
	w=0.75	200	4.98(0.01)	0.12(0.04)	97	7.45(0.20)
		400	4.98(0.01)	0.11(0.04)	97	7.93(0.20)
	w=0.90	200	4.95(0.02)	0.18(0.05)	94	8.60(0.20)
		400	4.94(0.02)	0.19(0.05)	94	9.11(0.30)
SCAD-SVM	w=0.50	200	4.96(0.01)	1.52(0.15)	96	8.01(0.20)
		400	4.96(0.01)	1.76(0.16)	96	8.36(0.20)
	w=0.75	200	4.88(0.03)	1.77(0.16)	92	9.59(0.30)
		400	4.82(0.04)	1.98(0.18)	92	10.27(0.40)
	w=0.90	200	4.82(0.04)	2.89(0.36)	90	11.32(0.50)
		400	4.77(0.04)	3.11(0.40)	89	11.87(0.40)
MCP-SVM	w=0.50	200	5.00(0.00)	0.27(0.02)	98	7.32(0.20)
		400	5.00(0.00)	0.29(0.02)	98	7.41(0.20)
	w=0.75	200	4.92(0.01)	0.43(0.03)	93	8.96(0.20)
		400	4.91(0.01)	0.47(0.03)	93	9.29(0.30)
	w=0.90	200	4.85(0.03)	0.88(0.05)	89	10.63(0.40)
		400	4.84(0.03)	0.91(0.050)	89	11.79(0.40)

Table 5.4: *Simulation outcomes for non-penalized SVMs with sample size* $n = 400$. *Margins are in brackets.*

Method	Proportion	p	Relevant	Irrelevant	True%	Test Error%
Adapt L_1-SVM	w=0.50	200	4.88(0.02)	2.42(0.66)	77	11.42(0.50)
		400	4.82(0.02)	2.65(0.23)	60	12.91(0.50)
	w=0.75	200	4.73(0.04)	3.69(0.30)	65	12.51(0.50)
		400	4.49(0.06)	3.82(0.23)	48	13.80(0.50)
	w=0.90	200	4.46(0.06)	5.52(0.63)	47	15.23(0.50)
		400	4.33(0.07)	6.18(0.76)	29	16.45(0.60)
L_1-SVM	w=0.50	200	4.49(0.08)	11.28(0.90)	35	13.28(0.50)
		400	4.25(0.9)	13.10(0.86)	16	16.55(0.60)
	w=0.75	200	4.25(0.09)	13.65(1.05)	17	15.97(0.50)
		400	4.12(0.09)	14.16(1.03)	6	18.46(0.60)
	w=0.90	200	4.09(0.10)	14.85(1.22)	5	18.98(0.60)
		400	4.01(0.10)	15.26(1.21)	1	21.98(0.70)
L_0-SVM	w=0.50	200	4.88(0.04)	25.08(1.22)	15	14.91(0.40)
		400	4.79(0.06)	28.66(1.56)	8	17.76(0.50)
	w=0.75	200	4.65(0.07)	28.12(1.54)	5	16.53(0.50)
		400	4.45(0.08)	31.67(1.53)	1	20.35(0.70)
	w=0.90	200	4.43(0.09)	33.53(1.61)	0	19.53(0.60)
		400	4.11(0.09)	40.27(2.08)	0	23.16(0.90)

Table 5.5: *Classification Outcome on the Wisconsin Breast Cancer Data Set. Margins are provided in the brackets.*

Methods	# of Predictors	Prediction Error(%)
DA-SCAD-SVM	6.05(0.80)	9.60(0.30)
DA-MCP-SVM	5.06(0.20)	9.40(0.20)
SCAD-SVM	7.10(0.80)	10.90(0.30)
MCP-SVM	6.04(0.20)	13.20(0.20)
L_0-norm Approximation SVM	12.04(1.30)	15.20(0.20)
Adapt L_1-norm SVM	14.50(2.40)	17.20(1.50)

Chapter 6

Conclusions And Future Work

6.1 Conclusions and Discussions

The motivation of the thesis comes from a real projection in how to classify a patient whether he or she has a caner, some other disease or no disease in prostate in Canada, based on various measures of medical imaging data that come from different platforms including MRI, ultrasound and CT. The predictive models that are derived from statistical learning in this work can help automatically predict the healthiness of a patient based on medical images only, much less time-consuming and more accessible to those patients with less medical resources.

Compared with the traditional classification problem, data are always imbalanced due to the fact that even a patient with a very severe cancer will only show 10% to 20% cancerous proportions, which lead to the poor performance in traditional classification methods. Another issue is that how to determine the true measures of the medical images that can have real relation to the disease, instead of involving redundant or even noisy features from the medical images, which might takes extra cost both in time and money.

The Support Vector Machine (SVM) is one of the most popular used algorithms on classification, and are well renowned for their strong theoretical foundations, gen-

eralization performance and ability to handle high dimensional data. By mapping the training data into a higher-dimensional feature space implicitly, SVM can construct a hyperplane (decision surface) in this feature space and maximize the margin of separation between itself and those points locating nearest to it (called the Support Vectors (SV)). Then, this decision surface can then be used as the rule and basis for classifying vectors of unknown objects.

Considering all these factors, our main contributions are in the following. We present a new method of scaling the kernel function to make it data-adaptive, consisting of two steps. Followed by the primary SVM procedure in the first step, the method locally adapts the kernel function to the data distribution based on the skewness of the class boundary and hence enlarge the kernel directly on the Riemannian manifold in the feature space, instead of the positions of support vectors in the input space. With the distance measure in the feature space, the conformal transformation can make full use of the updated information in the first step. Experimental results on both simulation and MRI data show that this new way of constructing the kernel is robust and performs well compared with the original method.

Furthermore, we extend our data-dependent SVM construction technique to multi-class situation, especially for the imbalanced data set. The algorithm still consists of two stages. Based on the idea of data-adaptive kernel SVM for binary case, In the first stage, a standard multi-class SVM with direct method is constructed so that the spatial location of all the support vectors can be found. Based on this, in the second stages, the data-dependent kernels are constructed for each SVMs in multi-class, combing the spatial location of the support vectors from the first stage and the information from sample sizes. By enhancing the local magnification effect, the separation of the SVMs with the data-adaptive kernels constructed in this way are more effective and robust, with the magnification effect varying along with the density of the size of neighbours, especially for imbalanced data. Thus, the kernel can be adapted optimally for a specific data set. Numerical results have shown the out-performance

with our methods.

We propose the data-adaptive kernel-penalized SVM, a new method that simultaneously achieves classification and feature selection, especially when the data is imbalanced. Instead of penalizing the loss function of SVMs, as has been done in literature, an non-convex penalty is proposed to be added directly to the kernel form of the SVM. The benefit that, the data-adaptive kernel is applicable to SVM so that even when the data is imbalanced, the performance of the SVM is still excellent, while in this setting other penalized SVM cannot work well due to lack of flexibility in SVM. This is because it is the kernel function that mainly determines the classification process. Along with the oracle properties as if the true sparsity in the feature space is already known, our proposed method works well in both simulation study and the real data example, possibly even when the ultra-dimensional setting exists.

6.2 Future Work

6.2.1 Measurement Error from Multiple Platforms

Measurement error is the difference between a measured value of some quantity and its true value. Statistically, an error is not a mistake; instead, it results in additional variability inherent in the measurement process. In particular, classification error is a type of measurement error by which the respondent does not provide a true response to an examination item. When measurement error comes from multiple platforms, the cumulative variability during each process can dramatically impact the performance of the classifier.

In imaging data process, there is measurement error from multiple platforms. For example, when medical clinics collect imaging data from patients with cancer, data come separately from different sources, such as Magnetic resonance imaging (MRI),

ultrasound or CT. When classifying different tissues according to different levels of severity of cancer, co-registration of different platforms and histology needs to be done for the sake of pathological reasons, and the accuracy of the classification is of extreme significance. However, when co-registering the In-Vivo MRI to Ex-Vivo MRI and then to CT data, transformation needs to be applied, so that the images for the same organs need to be overlapped, while during the process, additional variability is introduced by different operators. Besides, occasionally the measures of the predictors in medical science has to be recalculated by some threshold from raw imaging data due to real needs, the measurement error is further generated into the data. Thus, the multi-platform measurement error has to be considered so that the performance of the classifier is reliable.

It is possible to take the measure error into consideration in our settings. When the data is polluted by measure error, the performance of the classifier can be dramatically impacted and thus the prediction power. Intuitively, SVM should be robust and resistant to measurement error from weak to moderate levels, as the decision boundary relies only on the sample points near the boundary rather than the observations far away. Thus, even if there is some measurement error in the data, the estimated decision boundary will be less affected.

6.2.2 Feature Engineering with Multi-class Classification

A potential application of our data-adaptive SVM is in feature engineering, especially in image recognition. When to detect the pattern in images in engineering, such as the edge of the prostate cancer, the spatial correlation needs to be considered due to the possibly high correlation between neighbourhood in some areas, and the classification process should take into the spatial information into account. A typical way to detect the patterns in images is to create some feature vectors from the predictors, such as the gradients in different directions (typically in horizontal and vertical directions) in the colour system pixel-wisely, where the gradients are derived from some specific

intensity measure. Thus, if these information can be used and incorporated into the data-adaptive kernel, the spatial information can be further used to detect different classes of the pixels and hence accuracy can be increased.

Another issue during the feature detection process is that, if we would like to detect labels of some area instead of pixel-wisely, more gradient features will be engages. More straightforwardly, if we want to classify a k by k region of pixels in an image with p intensity measure associated with each pixels, there will be k^2 gradients in horizontal and vertical directions, along with $k * p$ intensity measures. Thus, the number of potential predictors might increase proportionally with the area of the interested regions. When all these predictors are incorporated into the models, feature selection in our proposed method will come in to select the real useful predictors and eliminate the noisy ones.

6.2.3 Application to Multi-label Classification

Another potential application of our proposed data-adaptive SVM and feature selection method can is in multi-label classification problem, where an object is not exclusively belongs to a specific class, but rather might belong to more than one labels. An active principled approach is the active learning, which aims to optimize the performance of the classification while minimizing the number of labels needed for training. SVM has already been applied in this area, and the multi-class data-adaptive kernel machine in our method might also increase the separability of labeling process, especially when the data is highly imbalanced, which is quite common in active learning approach, though the data-adaptive kernel needs to be modified to incorporate with the setting. Real applications can be found in email documentation and library management areas.

6.2.4 Application to Unsupervised and Semi-supervised Learning

The technique of our proposed method might also be used in unsupervised and semi-supervised learning. Though the SVM is originally designed for supervised learning, it is also of help in un/semi-supervised learning. A typical way is to formulate convex relaxations of the natural training criterion: find a labeling process that can obtain an optimal SVM on the resulting training data, both in binary and multi-class problems, where the SVM gets involved. Thus, our proposed method, which considers the imbalanced data situation, can provide help from this perspective. Areas in bioinformatics such as gene selection have potentially its need in un/semi-supervised, such as clustering genes with similar functions and grouping the genes that achieve some specific function all together, where level of imbalance can be extremely high, and hence our proposed work can provide some insight.

Bibliography

Amari, S.-i. and Wu, S. (1999). Improving support vector machine classifiers by modifying kernel functions. *Neural Networks* **12,** 783–789.

Blake, C. L. and Merz, C. J. (1998). Uci repository of machine learning databases. *Department of Information and Computer Science, University of Califonia* .

Blum, A. L. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence* **97,** 245–271.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* **5,** 144–152.

Bradley, P. S. and Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. *Proceedings of the Fifteenth International Conference on Machine Learning* **15,** 82–90.

Bredensteiner, E. J. and Bennett, K. P. (1999). Multicategory classification by support vector machines. *Computational Optimization and Applications* **12,** 53–80.

Cawley, G. C. and Talbot, N. L. (2007). Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters. *Journal of Machine Learning Research* **8,** 841–861.

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

Chen, J. and Chen, Z. (2008). Extended bayesian information criteria for model selection with large model spaces. *Biometrika* **95,** 759–771.

Christmann, A. and Hable, R. (2012). Consistency of support vector machines using additive kernels for additive models. *Computational Statistics & Data Analysis* **56,** 854–873.

Claeskens, G., Croux, C., and Kerckhoven, J. V. (2008). An information criterion for variable selection in support vector machines. *Journal of Machine Learning Research* **9,** 541–558.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning* **20,** 273–297.

Crammer, K. and Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research* **2,** 265–292.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., Weingessel, A., and Leisch, M. F. (2006). The e1071 package. *Misc Functions of Department of Statistics (e1071), TU Wien* .

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association* **96,** 1348–1360.

Farquhar, J., Hardoon, D., Meng, H., Shawe-taylor, J. S., and Szedmak, S. (2006). Two view learning: svm-2k, theory and practice. *Advances in Neural Information Processing Systems* **1,** 355–362.

Fawcett, T. and Provost, F. (1997). Adaptive fraud detection. *Data Mining and Knowledge Discovery* **1,** 291–316.

Fumera, G. and Roli, F. (2002). Support vector machines with embedded reject option. *Pattern Recognition with Support Vector Machines* **2388,** 811–919.

Fung, G. M. and Mangasarian, O. L. (2005). Multicategory proximal support vector machine classifiers. *Machine Learning* **59,** 77–97.

Guermeur, Y. (2002). Combining discriminant models with new multi-class svms. *Pattern Analysis & Applications* **5,** 168–179.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research* **3,** 1157–1182.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning* **46,** 389–422.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York.

He, X., Wang, Z., Jin, C., Zheng, Y., and Xue, X. (2012). A simplified multi-class support vector machine with reduced dual optimization. *Pattern Recognition Letters* **33,** 71–82.

Izmailov, A. F. and Solodov, M. V. (2003). Karush-kuhn-tucker systems: regularity conditions, error bounds and a class of newton-type methods. *Mathematical Programming* **95,** 631–650.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning.* Springer New York.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. *European Conference on Machine Learning* **10,** 137–142.

Lee, Y., Lin, Y., and Wahba, G. (2004). Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association* **99,** 67–81.

Lin, Y. (2002). Support vector machines and the bayes rule in classification. *Data Mining and Knowledge Discovery* **6,** 259–275.

Maldonado, S. and Weber, R. (2009). A wrapper method for feature selection using support vector machines. *Information Sciences* **179,** 2208–2217.

Maldonado, S., Weber, R., and Basak, J. (2011). Simultaneous feature selection and classification using kernel-penalized support vector machines. *Information Sciences* **181,** 115–128.

Mangasarian, O. L. and Wild, E. W. (2001). Proximal support vector machine classifiers. In *Proceedings KDD-2001: Knowledge Discovery and Data Mining*. Citeseer New York.

Maratea, A. and Petrosino, A. (2011). Asymmetric kernel scaling for imbalanced data classification. *Fuzzy Logic and Applications* **9,** 196–203.

Maratea, A., Petrosino, A., and Manzo, M. (2014). Adjusted f-measure and kernel scaling for imbalanced data learning. *Information Sciences* **257,** 331–341.

Mazumder, R., Friedman, J. H., and Hastie, T. (2011). Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association* **106,** 1125–1138.

Mazurowski, M. A., Habas, P. A., Zurada, J. M., Lo, J. Y., Baker, J. A., and Tourassi, G. D. (2008). Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks* **21,** 427–436.

Menardi, G. and Torelli, N. (2014). Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery* **28,** 1–31.

Park, C., Kim, K.-R., Myung, R., and Koo, J.-Y. (2012). Oracle properties of scad-penalized support vector machine. *Journal of Statistical Planning and Inference* **142,** 2257–2270.

Platt, J. C., Cristianini, N., and Shawe-Taylor, J. (1999). Large margin dags for multiclass classification. *Neural Information Processing Systems* **12,** 547–553.

Sánchez A, V. D. (2003). Advanced support vector machines and kernel methods. *Neurocomputing* **55,** 5–20.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT press Cambridge.

Suykens, J. A., Van Gestel, T., and De Brabanter, J. (2002). *Least Squares Support Vector Machines.* World Scientific, Singapore.

Suykens, J. A. and Vandewalle, J. (1999a). Least squares support vector machine classifiers. *Neural Processing Letters* **9,** 293–300.

Suykens, J. A. and Vandewalle, J. (1999b). Multiclass least squares support vector machines. *International Joint Conference on Neural Network* **2,** 900–903.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* **58,** 267–288.

Tong, S. and Chang, E. (2001). Support vector machine active learning for image retrieval. *Proceedings of the Ninth Association for Computing Machinery International Conference on Multimedia* **9,** 107–118.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. *Proceedings of the Twenty-first International Conference on Machine Learning* **21,** 104.

van Stiphout, R. G., Lammering, G., Buijsen, J., Janssen, M. H., Gambacorta, M. A., Slagmolen, P., Lambrecht, M., Rubello, D., Gava, M., Giordano, A., O.Postma, E., Karin, H., Carlo, C., Vincenzo, V., and Philippe, L. (2011). Development and external validation of a predictive model for pathological complete response of rectal cancer patients including sequential pet-ct imaging. *Radiotherapy and Oncology* **98,** 126–133.

Vapnik, V. (2006). *Estimation of Dependences Based on Empirical Data.* Springer, New York.

Vapnik, V. N. and Vapnik, V. (1998). *Statistical Learning Theory.* Wiley, New York.

Wang, L. and Shen, X. (2007). On l1-norm multiclass support vector machines: methodology and theory. *Journal of the American Statistical Association* **102,** 583–594.

Wang, L., Zhu, J., and Zou, H. (2006). The doubly regularized support vector machine. *Statistica Sinica* **16,** 589–615.

Wang, L., Zhu, J., and Zou, H. (2008). Hybrid huberized support vector machines for microarray classification and gene selection. *Bioinformatics* **24,** 412–419.

Weston, J. and Watkins, C. (1998). Multi-class support vector machines. Technical report, *Department of Computer Science, Royal Holloway*, University of London.

Weston, J. and Watkins, C. (1999). Support vector machines for multi-class pattern recognition. *European Symposium on Artificial Neural Networks* **7,** 219–224.

Williams, P., Li, S., Feng, J., and Wu, S. (2005). Scaling the kernel function to improve performance of the support vector machine. *Advances in Neural Networks* **2,** 831–836.

Wu, G. and Chang, E. Y. (2003). Adaptive feature-space conformal transformation for imbalanced-data learning. *International Conference on Machine Learning* **20,** 816–823.

Wu, S. and Amari, S.-I. (2002). Conformal transformation of kernel functions: A data-dependent way to improve support vector machine classifiers. *Neural Processing Letters* **15,** 59–67.

Xia, X.-L. C. and Li, K. (2008). A sparse multi-class least-squares support vector machine. *IEEE International Symposium on Industrial Electronics* **28,** 1230–1235.

Xiang-min, X., Yun-Feng, M., Jia-Ni, X., and Feng-le, Z. (2007). Classification performance comparison between rvm and svm. *IEEE International Workshop on Anti-counterfeiting, Security, Identification* **27,** 208–211.

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics* **38,** 894–942.

Zhang, X., Wu, Y., Wang, L., and Li, R. (2016). Variable selection for support vector machines in moderately high dimensions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **78,** 53–76.

Zhang, Y., Fu, P., Liu, W., and Chen, G. (2014). Imbalanced data classification based on scaling kernel-based support vector machine. *Neural Computing and Applications* **25,** 927–935.

Zhou, Q., Lin, C., Luo, L., and Peng, H. (2007). An imbalanced training data svm classification problem based on riemannian metric. *Proceedings of the Twenty-sixth Chinese Control Conference* **26,** 554–557.

Zhu, J. and Hastie, T. (2005). Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics* **14,** 185–205.

Zhu, J., Rosset, S., Tibshirani, R., and Hastie, T. J. (2003). 1-norm support vector machines. *Advances in Neural Information Processing Systems* **16,** 49–56.

Zou, H. (2007). An improved 1-norm svm for simultaneous classification and variable selection. *Artificial Intelligence and Statistics* **2,** 675–681.

Zou, H. and Yuan, M. (2008). The $f_\infty-$norm support vector machine. *Statistica Sinica* **18,** 379–398.

Appendix A

Supplementary Materials

A.1 Mercer's Condition

In mathematics, a real-valued function $K(x, y)$ is said to fulfill Mercer's condition if for all square integrable function $g(x)$ one has

$$\int\int g(x)K(x,y)g(y)dxdy \geq 0. \tag{A.1}$$

which is a sufficient condition for every kernel function in SVM (Vapnik and Vapnik (1998)). In terms of the SVMs with data-adaptive functions, it can be proved that the updated "kernel" based on local data is indeed a kernel function that satisfies the Mercer's Condition (Wu and Amari (2002)).

A.2 An Explanation on the Geometry of SVM Kernels

Denote by $\mathbf{f}$ the mapped result of $\mathbf{x}$ in F, i.e., $\mathbf{f} = \mathbf{s(x)}$. Then a small change in $\mathbf{x}$, d$\mathbf{x}$, will be mapped to

$$\mathrm{d}\mathbf{f} \;=\; \nabla\mathbf{s} \cdot \mathrm{d}\mathbf{x} = \sum_j \frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\ \mathrm{d}x_j, \tag{A.2}$$

where

$$\nabla \mathbf{s} \;=\; \left(\frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\right). \tag{A.3}$$

Thus, the squared length of $d\mathbf{f}$ can be written in the quadratic form as

$$|d\mathbf{f}|^2 \;=\; \sum_{m=1}^{l}(d\mathbf{z}_m)^2 = \sum_{ij} s_{ij}(\mathbf{x})\mathrm{d}x_i\ \mathrm{d}x_j, \tag{A.4}$$

where

$$s_{ij}(\mathbf{x}) = \left(\frac{\partial}{\partial x_i}\mathbf{s}(\mathbf{x})\right)^T \cdot \left(\frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\right) \tag{A.5}$$

Then the $n \times n$ positive-definite matrix $S(\mathbf{x}) = (s_{ij}(\mathbf{x}))$ is defined on the Riemannian metric. It has been proved that the metric can be derived from the kernel K.

A.3 Proof of Amplification of Separability on Boundaries

Take the binary case as an example. With Theorem 3.3.2, it is easy to check the Euclidean metric is

$$s_{ij}(\mathbf{x}) \;=\; \frac{1}{\sigma^2}\eta_{ij} \tag{A.6}$$

and the volume magnification is the constant

$$\sqrt{s(\mathbf{x})} \;=\; \frac{1}{\sigma^n} \tag{A.7}$$

where $\eta_{ij} = I(i = j)$ and $I(\cdot)$ is the indicator function. Thus, the modified mapping $\tilde{s}_{ij}(\mathbf{x})$, corresponding to the conformally transformed kernel function $\tilde{K}$ when K is Gaussian RBF kernel, is

$$\tilde{s}_{ij}(\mathbf{x}) \;=\; \frac{|c(\mathbf{x})|}{\sigma^2}\eta_{ij} + c_i(\mathbf{x})c_j(\mathbf{x}) \tag{A.8}$$

where $c_i(\mathbf{x})$ is the corresponding component of $\nabla c(\mathbf{x}) = c(\mathbf{x})\nabla \log\, c(\mathbf{x})$, thus the ratio of the new to the old magnification factors turns out by

$$\sqrt{\frac{\tilde{s}(\mathbf{x})}{s(\mathbf{x})}} \;=\; c^n(\mathbf{x})\sqrt{1 + \sigma^2\|\nabla \log\, c(\mathbf{x})\|^2} \tag{A.9}$$

where $\log c(\mathbf{x}) = -k_M|D(\mathbf{x})|$, and thus Eq.(A.9) is changed into

$$\sqrt{\frac{\tilde{s}(\mathbf{x})}{s(\mathbf{x})}} \;=\; e^{-nk_M|D(\mathbf{x})|}\sqrt{1+4\sigma^2k_M^2|D(\mathbf{x})|\|\nabla|D(\mathbf{x})|\|}. \tag{A.10}$$

Thus, the magnification will be at least $e^{-nM}\sqrt{1+4M^2\sigma^2}$ at the separating region where $D(\mathbf{x}) = \pm 1$, since $k_M < M$ by its definition.

Appendix B

Proofs of Theorems

B.1 Proof of Result 3.3.1

Proof. By the definition of a reproducing kernel function $K(\mathbf{x}, \mathbf{z})$ with its values λ_k and the corresponding scalar eigenfunctions $\mathrm{g}_k(\mathbf{x})$, we have

$$\int K(\mathbf{x}, \mathbf{z}) \cdot \mathrm{g}_k(\mathbf{z}) \, \mathrm{d}\mathbf{z} \;=\; \lambda_k \cdot \mathrm{g}_k(\mathbf{x})$$

where $k = 1, 2, \ldots, l$. Then the kernel is represented as

$$K(\mathbf{x}, \mathbf{z}) \;=\; \sum_k \lambda_k \cdot \mathrm{g}_k(\mathbf{x}) \cdot \mathrm{g}_k(\mathbf{z}).$$

By rescaling the function $g_k(\cdot)$ as $s_k(\mathbf{x}) = \sqrt{\lambda_k} g_k(\mathbf{x})$, the kernel function can be further present as

$$K(\mathbf{x}, \mathbf{z}) \;=\; \sum_k \mathrm{s}_k(\mathbf{x}) \cdot \mathrm{s}_k(\mathbf{z}) \;=\; [\mathbf{s}(\mathbf{x})]^T \cdot [\mathbf{s}(\mathbf{z})]$$

where $[\mathbf{s}(\mathbf{x})]^T = \big(\mathrm{s}_1(\mathbf{x}), \mathrm{s}_2(\mathbf{x}), \ldots, \mathrm{s}_l(\mathbf{x})\big)$ and $[\cdot]^T$ is the transpose operator. Thus, if we further define

$$\nabla \mathbf{s} \;=\; \left(\frac{\partial\, \mathbf{s}(\mathbf{x})}{\partial\, \mathbf{x}}\right) = \begin{pmatrix} \frac{\partial\ s_1(\mathbf{x})}{\partial\ x_1} & \cdots & \frac{\partial\ s_1(\mathbf{x})}{\partial\ x_p} \\ \vdots & \vdots & \vdots \\ \frac{\partial\ s_l(\mathbf{x})}{\partial\ x_1} & \cdots & \frac{\partial\ s_l(\mathbf{x})}{\partial\ x_p} \end{pmatrix}$$

and

$$\begin{aligned} \mathrm{s}_{ij}(\mathbf{x}) &= \left(\frac{\partial}{\partial x_i}\mathbf{s}(\mathbf{x})\right)^T \cdot \left(\frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\right) \\ &= \left(\frac{\partial \mathrm{s}_1(\mathbf{x})}{\partial x_i}, \ldots, \frac{\partial \mathrm{s}_l(\mathbf{x})}{\partial x_i}\right) \cdot \left(\frac{\partial \mathrm{s}_1(\mathbf{x})}{\partial x_j}, \ldots, \frac{\partial \mathrm{s}_l(\mathbf{x})}{\partial x_j}\right)^T, \end{aligned}$$

as in Eq.(3.3.1) and (3.5), it follows that

$$\frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}K(\mathbf{x},\mathbf{z})|_{\mathbf{z}=\mathbf{x}} = [\nabla \mathbf{s}(\mathbf{x})]^T \cdot \nabla \mathbf{s}(\mathbf{z}) = \left(\frac{\partial}{\partial x_i}\mathbf{s}(\mathbf{x})\right)^T \cdot \left(\frac{\partial}{\partial x_j}\mathbf{s}(\mathbf{x})\right) = \mathrm{s}_{ij}(\mathbf{x}). \quad \sharp$$

□

The lemma gives how a mapping $\mathbf{s}$ is associated with the corresponding kernel function K. Thus, given a specific form of a kernel function and an adaptive scaling function $c(\mathbf{x})$, we have the theorems 3.3.2 and 3.3.3.

B.2 Proof of Result 3.3.2

Proof. Assume the primary kernel function as $K(\mathbf{x},\mathbf{z})$ and a scalar function $c(\mathbf{x})$ as in Eq.(3.7). If we define

$$\begin{aligned} c_i(\mathbf{x}) &= \frac{\partial c(\mathbf{x})}{\partial x_i}, \\ K_{i\cdot}(\mathbf{x},\mathbf{x}) &= \frac{\partial K(\mathbf{x},\mathbf{z})}{\partial x_i}|_{\mathbf{z}=\mathbf{x}}, \\ K_{\cdot j}(\mathbf{x},\mathbf{x}) &= \frac{\partial K(\mathbf{x},\mathbf{z})}{\partial z_j}|_{\mathbf{z}=\mathbf{x}}, \\ \text{and} \qquad \mathrm{s}_{ij}(\mathbf{x}) &= \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}K(\mathbf{x},\mathbf{z})|_{\mathbf{z}=\mathbf{x}} \end{aligned}$$

then by Lemma 3.3.1,

$$\begin{aligned}
\tilde{\mathrm{s}}_{ij}(\mathbf{x}) &= \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}\tilde{K}(\mathbf{x},\mathbf{z})|_{\mathbf{z}=\mathbf{x}} = \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}\left(c(\mathbf{x})\cdot K(\mathbf{x},\mathbf{z})\cdot c(\mathbf{z})\right)|_{\mathbf{z}=\mathbf{x}} \\
&= \frac{\partial}{\partial z_j}\left[c(\mathrm{z})\cdot\left(\frac{\partial}{\partial x_i}\left[c(\mathbf{x})\cdot K(\mathbf{x},\mathbf{z})\right]\right)\right]\Bigg|_{\mathbf{z}=\mathbf{x}} \\
&= \frac{\partial}{\partial z_j}\left[c(\mathrm{z})\cdot\left(c_i(\mathbf{x})\cdot K(\mathbf{x},\mathbf{z}) + c(\mathbf{x})\cdot\frac{\partial}{\partial x_i}K(\mathbf{x},\mathbf{z})\right)\right]\Bigg|_{\mathbf{z}=\mathbf{x}} \\
&= c_i(\mathbf{x})\cdot c_j(\mathbf{z})\cdot K(\mathbf{x},\mathbf{z}) + c_i(\mathbf{x})\cdot c(\mathbf{z})\cdot\frac{\partial}{\partial z_j}K(\mathbf{x},\mathbf{z}) + c_i(\mathbf{x})\cdot c_j(\mathbf{z})\frac{\partial}{\partial x_i}K(\mathbf{x},\mathbf{z}) \\
&\quad + c(\mathbf{x})\cdot c(\mathbf{z})\cdot\frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}K(\mathbf{x},\mathbf{z})\Bigg|_{\mathbf{z}=\mathbf{x}} \\
&= c^2(\mathbf{x})\cdot\mathrm{s}_{ij}(\mathbf{x}) + c_i(\mathbf{x})\cdot K(\mathbf{x},\mathbf{x})\cdot c_j(\mathbf{x}) + c(\mathbf{x})\cdot\{K_{i\cdot}(\mathbf{x},\mathbf{x})\cdot c_j(\mathbf{x}) + K_{\cdot j}(\mathbf{x},\mathbf{x})\cdot c_i(\mathbf{x})\}.
\end{aligned}$$

In particularly when $i = j$, it is easy to check that

$$K_{i\cdot}(\mathbf{x},\mathbf{x}) = K_{\cdot j}(\mathbf{x},\mathbf{x}),$$

and define them in short as $K_i(\mathbf{x},\mathbf{x})$, it follows that

$$\tilde{\mathrm{s}}_{ii}(\mathbf{x}) = c^2(\mathbf{x})\cdot\mathrm{s}_{ii}(\mathbf{x}) + 2\cdot c(\mathbf{x})\cdot c_i(\mathbf{x})\cdot K_i(\mathbf{x},\mathbf{x}) + c_i^2(\mathbf{x})\cdot K(\mathbf{x},\mathbf{x}). \quad \sharp$$

□

Thus, given a specific form of the primary kernel function K, the adaptive scaling mapping can be calculated. When the Gaussian RBF kernel is applied, we have Theorem 3.3.3, as proved in the following.

B.3 Proof of Result 3.3.3

Proof. When we apply in Theorem 3.3.2 the Gaussian RBF kernel as in (3.3.3), it is found that

$$K_{i\cdot}(\mathbf{x},\mathbf{x}) = K_{\cdot j}(\mathbf{x},\mathbf{x}) = 0$$

and

$$K(\mathbf{x},\mathbf{x}) = 1$$

for any i and j, so the third term in the result of Theorem 3.3.2 is 0, and second term is changed into $c_i(\mathbf{x}) \cdot c_j(\mathbf{x})$. Further, when $i \neq j$,

$$\mathrm{s}_{ij}(\mathbf{x}) = \frac{\partial}{\partial x_i}\frac{\partial}{\partial z_j}K(\mathbf{x},\mathbf{z})|_{\mathbf{z}=\mathbf{x}} = \frac{1}{\sigma^2}(x_i - z_i)\cdot K(\mathbf{x},\mathbf{x})\cdot(x_j - z_j)\Big|_{\mathbf{z}=\mathbf{x}} = 0;$$

while when $i = j$,

$$\mathrm{s}_{ii}(\mathbf{x}) = \frac{1}{\sigma^2}\Big((x_i - z_i)\cdot K(\mathbf{x},\mathbf{z})\cdot(x_i - z_i) + K(\mathbf{x},\mathbf{z})\Big)\Big|_{\mathbf{z}=\mathbf{x}} = \frac{1}{\sigma^2};$$

thus, the first term becomes

$$\frac{c^2(\mathbf{x})}{\sigma^2}\cdot(i=j).$$

Combining all the above results, Theorem 3.3.3 is proved. ♯

□

B.4 Assumptions for Penalty Terms

1. The penalty function $p_{\lambda_n}(x)$ is symmetric, non-decreasing and concave for $x \in [0,\infty)$, with a continuous first-order derivative $p'_{\lambda_n}(x)$ on R^+ and $p'_{\lambda_n}(0) = 0$.

2. There exists $a > 1$, such that $\lim_{x\to 0+} p'_{\lambda_n}(x) = \lambda_n$, $p'_{\lambda_n}(x) \geq \lambda_n - x/a$ for $0 < x < a\lambda$ and $p'_{\lambda_n}(x) = 0$ for $x \geq a\lambda$.

B.5 Proof of Theorem 5.3.1: the Oracle Property in data-adaptive Kernel-penalized SVM

B.5.1 Regularity Conditions

1. The densities of $\mathbf{Z}$ given $Y = 1$ and $Y = -1$ are continuous with common support in R^q. Here $\mathbf{Z}$ is the truly active predictors.

2. $E(Z_j^2) < \infty$ for $1 \leq j \leq q$, i.e., the second order moments of all active predictors are finite.

3. The true parameter $\mathbf{w}_0$ is a non-zero and unique vector.

4. $q = O(n^c)$ for some $0 \leq c < 1/2$, namely, $\lim_{n\to\infty} q/n^c < \infty$.

5. The largest eigenvalue of $n^{-1}[\mathbf{X}^{\odot 2}]^T\mathbf{X}^{\odot 2}$ is finite, where $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_l)$ is design matrix, and $(\cdot)^{\odot 2}$ is the component-wise square.

Here condition 1-3 are the assumptions to ensure that the oracle estimator constructed in our proposed method is consistent and that the optimal classification decision rule is not constant. Condition 4 is a common requirement in high-dimensional inference, indicating that the the number of the truly active predictors cannot diverge with a rate faster than $\sqrt{n}$. Condition 5 gives the upper boundary of the largest eigenvalues of the squared design matrix, which is necessary in our proposed method due to the quadratic form in the radial kernel functions. With these conditions, the oracle property can be proved.

B.5.2 Proof of Theorem 5.3.1: the Oracle Property

Proof. Define

$$L(\mathbf{w}) = \sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j; \mathbf{w}) + \sum_{j=1}^{p} p_{\lambda_n}(|\mathrm{w}_j|) \;=\; L_1(\mathbf{w}) + \sum_{j=1}^{p} p_{\lambda_n}(|\mathrm{w}_j|) \quad \text{(B.1)}$$

that comes from the second part of the optimization problem in Eq.(5.12). We shall show that, for $\forall\ \epsilon > 0$, there is a constant Δ such that, when n is sufficiently large,

$$\mathbf{Pr}[\inf_{\|\mathbf{u}\|=\Delta} L(\mathbf{w}_{true} + \sqrt{q/n}\cdot\mathbf{u}) > L(\mathbf{w}_{true})] \;\geq\; 1-\epsilon. \quad \text{(B.2)}$$

In the following proof, $\mathbf{w}_{true}$ will be replaced by $\mathbf{w}$ for short without misleading the proof. Note that $\sum_{i=1}\alpha_i y_i = 0$ from the constraints of Eq.(5.11).

$$\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j = \sum_{i=1}^{l}\alpha_i y_i \cdot \sum_{j=1}^{l}\alpha_j y_j = 0, \quad \text{(B.3)}$$

and further,

$$0 = \sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j = \sum_{i,\ j,\ y_i y_j=1}\alpha_i\alpha_j - \sum_{i,\ j,\ y_i y_j=1}\alpha_i\alpha_j. \tag{B.4}$$

This immediately leads to

$$L_1(\mathbf{w}) \ = \sum_{i,\ j,\ y_i y_j=1}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w}) \ - \sum_{i,\ j,\ y_i y_j=1}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w}) \tag{B.5}$$

Since $y_i \in \{1,-1\}$, then $y_i y_j \in \{1,-1\}$ for all (i,j), with a probability of $\pi_+^2+\pi_-^2$ for 1 and $2\pi_+\pi_1$ for -1 assuming independence between y_i and y_j, where $\pi_+ = \Pr(y_i=1)$ and $\pi_- = \Pr(y_i=1)$, and further, it is easy to check

$$0 \le E(y_i y_j) = \pi_+^2+\pi_-^2-2\pi_+\pi_1 = (\pi_+-\pi_-)^2 \le 1$$

and thus

$$E(L_1(\mathbf{w})) \ = \ (\pi_+-\pi_-)^2\sum_{i=1}\sum_{j=1}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w}) \ge 0 \tag{B.6}$$

Now, let

$$\begin{aligned}
\Lambda_n(\mathbf{u}) = \ & nq^{-1}\cdot[L_1(\mathbf{w}+\sqrt{q/n}\cdot\mathbf{u}) - L_1(\mathbf{w})] \\
= \ & nq^{-1}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\exp\{-\frac{1}{2}q/n\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}-1\} \\
= \ & nq^{-1}\cdot\sum_{i,\ j,\ y_i\cdot y_j=1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\exp\{-\frac{1}{2}q/n\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}-1\} \\
& - nq^{-1}\cdot\sum_{i,\ j,\ y_i\cdot y_j=-1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\exp\{-\frac{1}{2}\sqrt{q/n}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}-1\}
\end{aligned} \tag{B.7}$$

where $()^{\odot 2}$ is the component-wise square. Since $\exp(x) > x+1$ for all x and $\alpha_i \ge 0$ for all i, then the first item in Eq.(B.7)

$$\begin{aligned}
& \ge nq^{-1}\sum_{i,\ j,\ y_i\cdot y_j=1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot[-\frac{1}{2}\sqrt{q/n}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}-1+1] \\
& = \sqrt{nq^{-1}}\sum_{i,\ j,\ y_i\cdot y_j=1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\{-\frac{1}{2}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}\};
\end{aligned} \tag{B.8}$$

Take standard augment on the Taylor expansion w.r.t. $\mathbf{u}$,

$$\exp\{-\frac{1}{2}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}-1\} = -\frac{1}{2}\sqrt{q/n}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u} + \frac{1}{4}\cdot\frac{q}{n}\cdot\mathbf{u}^T[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}][(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}+o_p(n^{-1}) \tag{B.9}$$

Then it is easy to find that the second item in Eq.(B.7) is

$$\leq nq^{-1}\cdot\sum_{i,\ j,\ y_i\cdot y_j=-1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot(-\frac{1}{2}\sqrt{q/n}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u} + \frac{1}{4}\cdot\frac{q}{n}\cdot\mathbf{u}^T[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}][(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}+o_p(1) \tag{B.10}$$

Now, by combining Eq.(B.8) and Eq.(B.10), we have

$$\begin{aligned}\Lambda_n(\mathbf{u}) \geq{}& \sqrt{nq^{-1}}\cdot[\sum_{i,\ j,\ y_i\cdot y_j=1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\{-\frac{1}{2}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}\\ &-\sum_{i,\ j,\ y_i\cdot y_j=-1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\{-\frac{1}{2}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}\}]\\ &+\frac{1}{4}\cdot\mathbf{u}^T[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}][(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}+o_p(1)\\ ={}&\sqrt{nq^{-1}}\cdot\sum_{i,j}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\{-\frac{1}{2}\cdot[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\}\cdot\mathbf{u}\\ &+\frac{1}{4}\sum_{i,\ j,\ y_i\cdot y_j=-1}^{l}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j;\mathbf{w})\cdot\mathbf{u}^T[(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}][(\mathbf{x}_i-\mathbf{x}_j)^{\odot 2}]^T\mathbf{u}+o_p(1)\end{aligned} \tag{B.11}$$

Note that the first part in Eq.(B.11) is equivalent to $\frac{\partial}{\partial\mathbf{w}} = L_1'(\mathbf{w}) = 0$ due to necessary condition that $\mathbf{w} = \arg\min L_1(\mathbf{w})$, and the second term, which is obviously non-negative, will dominate Eq.(B.11). In terms of the penalty term, it is obvious that

$$
\begin{aligned}
P_n(\mathbf{w}) &= nq^{-1}\sum_{j=1}^{p}[p_{\lambda_n}(|\mathrm{w}_j+\sqrt{q/n}\cdot u_j|-p_{\lambda_n}(|\mathrm{w}_j|)]; \quad \text{using } p_{\lambda_n}(0)=0 \text{ and } p_{\lambda_n}(\cdot)\geq 0 \\
&\geq \sum_{j=1}^{q} nq^{-1}\cdot[p_{\lambda_n}(|\mathrm{w}_j+\sqrt{q/n}\cdot u_j|-p_{\lambda_n}(|\mathrm{w}_j|)]; \quad \text{using Taylor Expansion} \\
&= \sum_{j=1}^{q}\cdot[q^{-1/2}p'_{\lambda_n}(|\mathrm{w}_j|)+p''_{\lambda_n}(|\mathrm{w}_j|)u_j^2\{1+o_p(1)\}],
\end{aligned}
\tag{B.12}
$$

which is bounded by $q^{-1/2}\|\mathbf{u}\|+\max\{|p''_{\lambda_n}(\mathrm{w}_j)|: \ \mathrm{w}_j\neq 0\}\|\mathbf{u}|$. Thus, by choosing a sufficiently large Δ, $P_n(\mathbf{w})$ is dominated by the second item in Eq.(B.11) as well. Thus, $L(\mathbf{w})=\Lambda_n(\mathbf{u})+P_n(\mathbf{w})$ is dominated by a non-negative item with probability 1 within a ball. This indicates that with a probability at least $1-\epsilon$, there exists a local minimum in the ball $\{\mathbf{w}+\sqrt{q/n}\cdot\mathbf{u}:\|\mathbf{u}\|\leq\Delta\}$, and hence there exists a local minimizer such that $\|\hat{\mathbf{w}}-\mathbf{w}\|=O_p\{\sqrt{q/n}\}$. Note that when the kernel function K is updated by $\tilde{K}$, nothing is changed except that there the kernel is multiplied by two finite constants constructed from the first stage of SVM, and hence the theorem still holds. This completes the proof. ♯

□

Curriculum Vitae

Name:	Xin Liu
Post-Secondary Education and Degrees:	University of Western Ontario London, ON 2012 - 2013 M.Sc. in Statistics University of Western Ontario London, ON 2013 - 2017 Ph.D. Candidate in Statistics
Related Work Experience:	Teaching Assistant Research Assistant The University of Western Ontario 2012 - 2017

Publications:

Liu, X and He, W. (2017). Adaptive scaling of the kernel function for the improvement of the support sector machine with an application to a prostate cancer study. Submitted to *Biometrics*.

Liu, X and He, W. (2017). Data-adaptive kernel-penalized support vector machine with an application to a breast cancer Study. Ready for submission.