FEATURE SELECTION AND MODEL SELECTION FOR SUPERVISED LEARNING ALGORITHMS

YANG JIAN BO (*M. Eng*)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2011

Acknowledgments

I give my deepest appreciation to Prof. Ong Chong-Jin who was guiding me on research during the last four years. His instructive suggestions, invaluable comments and discussions, constant encouragements and personal concerns greatly help me in every stage of my research. I am very respectful for his rigorous attitude of scholarship and diligence.

I acknowledge National University of Singapore provided finical support to me through Research Scholarship.

I also would like to thank my companions who generously help me in various ways during this research. Particularly, I owe sincere gratitude to Shen Kai-Quan, Wang Chen, Yu Wei-Miao, Sui Dan, Shao Shi-Yun, Wang Qing and other members in Mechatronics and Control Lab. These friends gave me lots of helps during the past few years in NUS. I am also grateful to technicians in Mechatronics and Control Lab for their facility support.

Finally, I want to express my sincere thanks to my family for their loves and special thanks to my wife Ju Li for making our life wonderful.

Table of Contents

Acknowledgments **i**

Summary **vi**

List of Tables **x**

List of Figures **xiii**

Acronyms **xiv**

Nomenclature **xv**

1 Introduction **1**

1.1 Background 3

1.1.1 Feature Selection 3

1.1.2 Model Selection 7

1.2 Motivations 9

1.3 Organization 11

2 Review **13**

2.1 Learning Methods 14

2.1.1 Support Vector Machine 14

2.1.2 Support Vector Regression 16

2.1.3 Entropy and Mutual Information 18

2.1.4 Bounds of Generalization Performance 19

2.2 Feature Selection Methods 21

2.2.1 Filter Methods 22

2.2.2 Wrapper Methods 27

2.3 Model Selection Methods 30

2.3.1 Grid Search Method 31

2.3.2 Gradient-based Methods 31

2.3.3 Regularization Solution Path of SVM 32

3 Feature Selection via Sensitivity Analysis of MLP Probabilistic Outputs **34**

3.1 Preliminary 35

3.2 The Proposed Wrapper-based Feature Ranking Criterion for Classification 37

3.3 Feature Selection Scheme 40

3.4 Numerical Experiment 42

3.4.1 Artificial Data Sets 43

3.4.2 Real-world Data Sets 48

3.4.3 Discussion 50

3.5 Summary 51

4 Feature Selection via Sensitivity Analysis of SVR Probabilistic Outputs 59

4.1 Preliminary 60

4.2 The Proposed Wrapper-based Feature Selection Criterion for Regression 62

4.3 Feature Selection Scheme 66

4.4 Numerical Experiment 67

4.4.1 Artificial Problems 69

4.4.2 Real Problems 71

4.4.3 Discussion 74

4.5 Summary 76

5 Feature Selection via Mutual Information Estimation 83

5.1 Preliminary 84

5.2 The Proposed Method 86

5.3 Connection with Other Methods 90

5.4 Numerical Experiment 94

5.4.1 Artificial Data Sets 95

5.4.2 Real Problem 99

5.4.3 Discussion 101

5.5 Summary 102

6 Determination of Global Minimum of Some Common Validation Function in Support Vector Machine 108

6.1 Preliminary 109

6.2 Finding the Global Optimal Solution 114

6.3 Numerical Experiment and Discussion 120

6.4 Summary 125

7 Conclusions 129

7.1 Contributions 129

7.2 Directions of Future Work 133

Bibliography 135

Appendices 147

Author's Publications 153

Summary

The thesis is concerned about feature selection and model selection in supervised learning. Specifically, three feature selection methods and one model selection method are proposed.

The first feature selection method is a wrapper-based feature selection method for multilayer perceptron (MLP) neural network. It measures the importance of a feature by the its sensitivity with respect to the posterior probability over the whole feature space. The results of experiments show that this method performs at least as well, if not better than the benchmark methods.

The second feature selection method is a wrapper-based feature selection method for support vector regressor (SVR). In this method, the importance of a feature is measured by the aggregation, over the entire feature space, of the difference of the output conditional density function provided by SVR with and without a given feature. Two approximations of this criterion are proposed. Some promising results are also obtained in experiments.

The third feature selection method is a filter-based feature selection method. It uses a mutual information based criterion to measure the importance of a feature in a backward

selection framework. Unlike other mutual information based methods, the proposed criterion measures the importance of a feature with the consideration of all features. As the results of numerical experiments show, the proposed method generally outperforms existing mutual information methods and can effectively handle the data set with interactive features.

The one model selection method is to tune the regularization parameter of support vector machine. The tuned regularization parameter by the proposed method guarantees the global optimum of widely used non-smooth validation functions. The proposed method highly relies on the solution path of SVM over a range of the regularization parameter. When the solution path is available, the computation needed is minimal.

List of Tables

3.1 The number of realizations that feature $1, 2$ are successfully ranked in the top two positions over 30 realizations for Weston Problem. 45

3.2 The number of realizations that optimal features are successfully ranked in the top four positions over 30 realizations for Corral Problems. 48

3.3 Description of real-world data sets for classification problems. 48

3.4 *t*-test on Abalone data set. 52

3.5 *t*-test on WBCD data set. 53

3.6 *t*-test on Wine data set. 55

3.7 *t*-test on Vehicle data set. 56

3.8 *t*-test on Image data set. 56

3.9 *t*-test on Waveform data set. 57

3.10 *t*-test on Hillvalley data set. 57

3.11 *t*-test on Musk data set. 58

4.1 The number of realizations that relevant feature are successfully ranked in the top positions over 30 realizations for three artificial problems. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold. 73

4.2 Description of real-world data sets for regression problem. 74

4.3 t-test on mpg data set. 77

4.4 t-test on abalone data set. 78

4.5 t-test on cputime data set. 79

4.6 t-test on housing data set. 80

4.7 t-test on pyrim data set. 81

4.8 t-test on triazines data set. 82

5.1 Description of Monk data sets 96

5.2 The number of realizations that feature $1,2,5$ are successfully ranked in the top three positions over 30 realizations for Monk-1 problem. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold. 96

5.3 The number of realizations that feature $2,4,5$ are successfully ranked in the top three positions over 30 realizations for Monk-3 problem. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold. 96

5.4 The number of realizations that feature $1,2$ are successfully ranked in the top two positions over 30 realizations for Weston problem. 98

5.5 Description of real-world data sets for classification. 102

5.6 Average time (sec) of yielding feature ranking lists by all methods over 30 realizations of real-world data sets. 103

5.7 t-test on Abalone data set. 105

5.8 t-test on WBCD data set. 105

5.9 t-test on Glass data set. 106

5.10 t-test on Wine data set. 106

5.11 t-test on Satimage data set. 107

5.12 t-test on Musk data set. 107

6.1 Pseudo Code 118

6.2 Characteristics of data sets used in the experiments. 125

6.3 Optimal λ value and 5-fold cross-validation error rates for GO, GRID-i and GRAD-i of the first realization. The smallest error rate for each data set is highlighted in bold. 126

6.4 Optimal λ value and Test error rates for GO, GRID-i and GRAD-i of the first realization. The smallest error rate for each data set is highlighted in bold. 127

6.5 Mean and Standard Deviations of $E^{\dagger}$ of GO, GRID-i and GRAD-i over the the 10 realizations. The smallest Mean for each data set is highlighted in bold. 128

List of Figures

1.1 Feature selection and model selection in a supervised learning task. The dashes box denotes the pre-processing procedure. 3

1.2 The framework of feature ranking. 4

1.3 Illustration on feature interacting effect. 6

1.4 Validation error rate for different values of $\lambda = C^{-1}$ for Sonar data set. 8

3.1 Architecture of softmax-based probabilistic MLP. 36

3.2 Average test error against top-ranked features over 30 realizations of Weston data sets for four training set sizes. 46

3.3 Average test error against top-ranked features over 30 realizations of three Corral data sets: (a) Corral-6. (b) Corral-46. (c) Corral-47. 49

3.4 Test error rates on Abalone data set 52

3.5 Test error rates on WBCD data set 52

3.6 Test error rates on Wine data set 53

3.7 Test error rates on Vehicle data set 53

3.8 Test error rates on Image data set 54

3.9 Test error rates on Waveform data set 54

3.10 Test error rates on HillValley data set 54

3.11 Test error rates on Musk data set 55

4.1 Demonstration of the proposed feature ranking criterion with $d = 1$. Dots indicate locations of y_i 64

4.2 Average MSE (left-hand side) and average SCC (right-hand side) against top-ranked features over 30 realizations for Exponential Function Problem with six different settings 72

5.1 Average test error against top-ranked features over 30 realizations of Monk-1 data sets for four training set sizes. 97

5.2 Average test error against top-ranked features over 30 realizations of Monk-3 data sets for four training set sizes. 97

5.3 Average test error against top-ranked features over 30 realizations of Weston data sets for five training set sizes. 99

5.4 Test error rates on Abalone data set 103

5.5 Test error rates on WBCD data set 103

5.6 Test error rates on Glass data set 104

5.7 Test error rates on Wine data set 104

5.8 Test error rates on Satimage data set 104

5.9 Test error rates on Musk data set 105

6.1 (a) Typical values of $\hat{\alpha}_i(\lambda), i \in \mathscr{E}(\lambda^\ell)$ for $\lambda^{\ell+1} < \lambda \leq \lambda^\ell$. (b) Typical values of $h_j(\lambda)$ for $\lambda^{\ell+1} < \lambda \leq \lambda^\ell$. Points A and B refer to two possible values of $h_j(\lambda^\ell)$, positive and negative. 116

6.2 Curves of cross-validation error rates (CVER) as functions of λ for data set svmguide3. Solid line - 5-fold CVER; Dashed line - smooth 5-fold CVER; Dashed-dot line - CVER of fold 1; Dot line - smooth CVER of fold 1. The CVER functions for the other folds are omitted to prevent clutter. The optimal λ is 0.114 or $\log_2(0.114) = -3.1329$. 123

6.3 The histogram of intervals having various values of $|\mathscr{I}_S^\ell|$ for the 5 folds of svmguide3 in the first realization. The set $|\bar{\Lambda}_k|$ for $k = 1$ to 5 are $630, 755, 727, 828$ and 754 respectively. 124

Acronyms

FSPP	Feature-based Sensitivity of Posterior Probability
KKT	Karush-Kuhn-Tucker
KL	Kullback-Leibler
LP	Linear Programming
MI	Mutual Information
MLP	Multi-layer Perceptron
MSE	Mean Squared Error
RFE	Recursive Feature Elimination
RP	Random Permutation
SCC	Squared Correlation Coefficient
SD	Sensitivity of Density function
SMO	Sequential Minimal Optimization
SVM	Support Vector Machine
SVR	Support Vector Regression
SVMpath	Entire Regularization Path of Support Vector Machine
ISVMP	Improved Regularization Path of Support Vector Machine

Nomenclature

A^T	transposed matrix (or vector)
A^{-1}	inverse matrix
C	regularization parameter, $C > 0$
$K(\cdot,\cdot)$	kernel function
I	identity matrix
$\mathbb{E}[\cdot]$	expectation of a random variable
$\mathscr{H}$	Hilbert space
$\mathscr{I}$	index set
$\mathbb{R}$	set of real numbers
$\mathbb{R}^d$	d-dimensional real Euclidean space
$\phi(\cdot)$	mapping function
$\mathscr{N}(\mu,\sigma)$	normal distribution with mean μ and variance σ^2
$\mathscr{D}$	a data set

x	an input vector, $x \in \mathbb{R}^d$
y	target value, $y \in \{1, \cdots, c\}$ or $y \in \mathbb{R}$ or $y \in \{\pm 1\}$
(x, y)	a pattern in $\mathscr{D}$
x_i^j	the value of the j-th feature of the i-th sample, $x_i^j \in \mathbb{R}$
ω_k	class k
d	dimension of input space
w	input weight vector or feature weight vector
b	bias or constant offset, $b \in \mathbb{R}$
λ	regularization parameter, $\lambda > 0$ and $\lambda = \frac{1}{C}$
$\boldsymbol{\alpha}, \boldsymbol{\alpha}^*$	column vectors of Lagrangian multipliers of SVM problems
α_i, α_i^*	Lagrangian multipliers
$\boldsymbol{\xi}, \boldsymbol{\xi}^*$	vectors of slack variables
ξ_i, ξ_i^*	slack variables
δ	the noise
$\delta_{i,j}$	the Kronecker delta
i, j, k	indices
$\|\cdot\|$	Euclidean norm

Chapter 1

Introduction

Machine learning is concerned with automatical prediction of unseen patterns based on known empirical data. Such a prediction is often encountered in various disciplines, such as computer vision, bioinfomatics, natural language processing, finance and medical applications. Based on desired outcomes of problems, machine learning algorithms can be broadly categorized into three paradigms: supervised learning, unsupervised learning and semi-supervised learning. Supervised learning is for the case where the labels of empirical data are given, for example, supervised classification and supervised regression. By contrast, unsupervised learning is for the case where the labels of empirical data are not provided. An example of this is clustering where data are clustered into several distinct groups. Semi-supervised learning is a compromise between supervised learning and unsupervised learning, in which a few labeled and a large amount of unlabeled data are available. Hence, semi-supervised learning can deal with both supervised and unsupervised learning problems: semi-supervised classification, regression

and clustering.

In this thesis, only supervised learning is considered. The goal of supervised learning algorithm is to infer the mapping $f: \mathscr{X} \rightarrow \mathscr{Y}$ between input space $\mathscr{X}$ and output space $\mathscr{Y}$ based on all the observed (i.e. empirical) input-output pairs $\{(x_i, y_i) | x_i \in \mathscr{X}, y_i \in \mathscr{Y}\}$, such that the resultant mapping has good performance on new unseen patterns. Besides developing an approximate of f, the success of a supervised learning algorithm often depends on the availability of informative input features, and the correct setting of the configuration of the algorithm. Their roles in a typical learning algorithm are depicted in Figure 1.1. Hence, feature selection and model selection can be seen as pre-processing procedures to a learning algorithm. The former yields the optimal input features while the latter yields the optimal hyperparameters to the learning algorithm. The common purpose of these two pre-processing procedures is to improve the generalization performance, i.e., the performance on unseen data, of the learning algorithm.

In the past few years, great success of feature selection and model selection for various learning algorithms have been achieved in bioinformatics, web mining, computer vision and other data mining fields [6, 20]. The content of this thesis focuses on these two areas under the supervised learning paradigm. It is worthy to note that they are also important in unsupervised and semi-supervised learning, but these issues are not considered in this thesis.

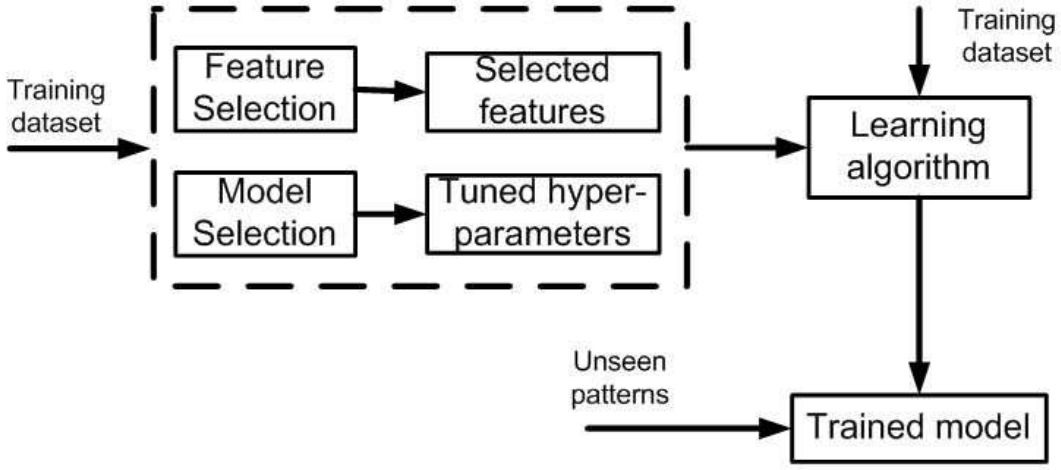


Figure 1.1: Feature selection and model selection in a supervised learning task. The dashes box denotes the pre-processing procedure.

1.1 Background

1.1.1 Feature Selection

Feature selection is a procedure of finding a set of most compact and informative original features [32, 31] for the purpose of predicting the output of the learning algorithm. In practice, many data sets have a huge number of features. For example, in the gene selection problems, the features are gene expression coefficients corresponding to the abundance of mRNA for a number of patients [31] and their number can range from 6,000 to 60,000. In text classification problems, the features are "bag of words" or vocabulary word frequency counts and can be hundreds of thousands in size. While having more features endows a learning algorithm with a greater discriminating power, performance degradation often sets in when many irrelevant or redundant features are included. The inclusion of irrelevant and redundant features also increases the computational complexity of the learning algorithm. Besides, it is also known that feature

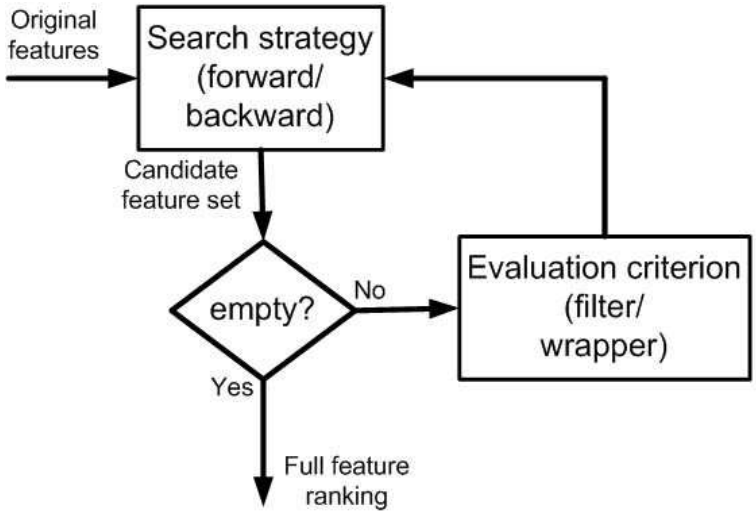


Figure 1.2: The framework of feature ranking.

selection can potentially benefit data visualization and data understanding, data storage reduction and the easy deployment of the learning algorithm. Consequently, feature selection has been an area of much research effort in various learning tasks [32, 33, 52].

If the input data have d features, there are a total of 2^d possible subsets of features. Obviously, it is not easy to directly select the desired features when d is large, although some efforts in this direction have been made [77, 90]. Many approaches choose feature ranking as an auxiliary mechanism to facilitate feature selection. The idea of feature ranking is to rank all features according to the importance of each feature. User can then select the desired number of features based on the resultant ranking list. As shown in Figure 1.2, the framework of feature ranking usually contains two constituents: feature evaluation criterion and subset search strategy.

A feature evaluation criterion measures the importance of a feature or a set of features and plays a crucial role in a feature selection method. The most direct evaluation criterion is the learning algorithm's prediction accuracy, as used in [70, 84]. However, its

implementation costs are typically very high for large data sets, since each evaluation requires training and predicting processes of the learning algorithm. In the past decades, various efficient evaluation criteria are proposed. Some of them rely on the learning algorithm with reduced training and predicting procedures. Methods that use learning algorithms are known as wrapper methods. By contrast, others are totally independent of the learning algorithm and only rely on the characteristics of the data set. These are known as filter methods.

A subset search strategy generates candidate feature subsets with the aim to find the optimal subset. The most direct search strategy is the exhaustive search, i.e., search among all possible feature subsets (2^d in total). As mentioned before, this is computationally intractable for data sets with many features. In practice, some heuristical search strategies are used: *forward* or *backward* search. Specifically, forward search begins with an empty set and successively adds one or a few most important features at each time, while backward search begins with a full set of features and successively removes one or a few least important features at each time [52].

Filter methods versus wrapper methods, and forward search versus backward search, which combination is the best? While it is still an open question [31, 32], some basic facts exist. In terms of computational efficiency, filter methods are faster than wrapper methods and forward search is faster than backward search in general. However, in terms of performance, filter methods and forward search have higher risk to suffer from performance degradation because of their limited capability to handle interacting effect of features.

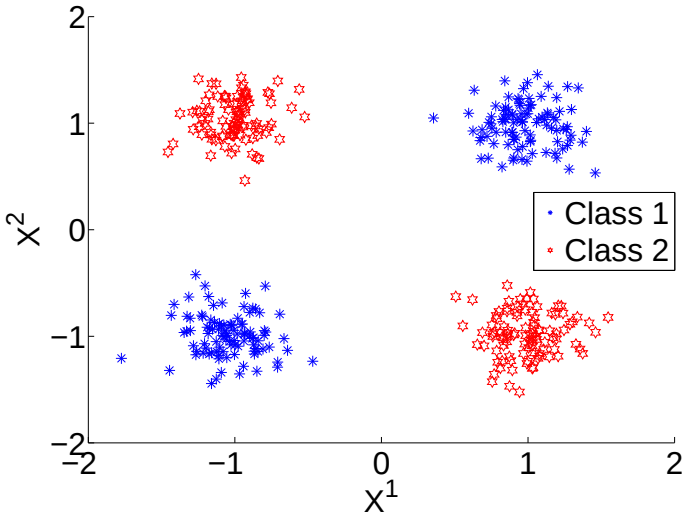


Figure 1.3: Illustration on feature interacting effect.

Interacting effect of features refers to the phenomenon that multiple variables that are useless individually can be useful together [31]. This phenomenon can be best illustrated by the famous "XOR" type problem as show in Figure 1.3. This figure shows a two class classification problem on a 2-dimensional data set, in which two Gaussian clumps are placed at the coordinates $(-1,-1),(1,1)$ for class 1 while another two are placed at $(1,-1),(-1,1)$ for class 2. Obviously, the projection of clumps on axis x^1 or x^2 leads to the perfect overlap of two classes and thus feature 1 and feature 2 are useless individually. But four clumps are well classified into two classes in the two dimensional space so features 1 and 2 are useful together.

Some filter methods assume that all features are independent and could not be able to handle the interacting effect well, while some forward methods (partially) ignoring the interacting effect also fail. These statements will be clarified and validated in the subsequent chapters.

1.1.2 Model Selection

Model selection refers to the procedure of tuning the hyperparameters of the learning algorithms. Hyperparameters ubiquitously exist in learning algorithms. For examples, in Multi-layer Perceptron (MLP) neural networks [5], hyperparameters include the number of layers and the number of hidden neurons. In Support Vector Machines (SVMs) [7, 81], hyperparameters include the regularization parameter and the kernel parameter. Different choices of these hyperparameters for learning algorithms can lead to drastically different performances [20, 35]. Hence, model selection is crucial for learning tasks and has been one active research topic [12, 19, 34, 45]. In this thesis, model selection is restricted on tuning the regularization parameter of SVM classifiers.

In 1992, Support Vector Machine (SVM) is first proposed for classification in the work [7]. Later, the principles underlying SVM are systematically developed in the framework of statistical learning theory by Vapnik [79, 81]. The extensions of SVM to regression, density estimation, clustering and structure output learning are proposed in [81, 78] and the references thereof. Today SVM is a well-known learning tool and several outstanding numerical routines of SVM have been developed [10, 41, 62, 44, 39, 34, 58].

Basically, SVM can be formulated into the following regularized empirical risk minimization form:

$$\min_{f} \ \Omega(f) + CR_{emp}(f) \tag{1.1}$$

where f is the predictor to to be learned, $R_{emp}(f)$ is the empirical loss on the observed

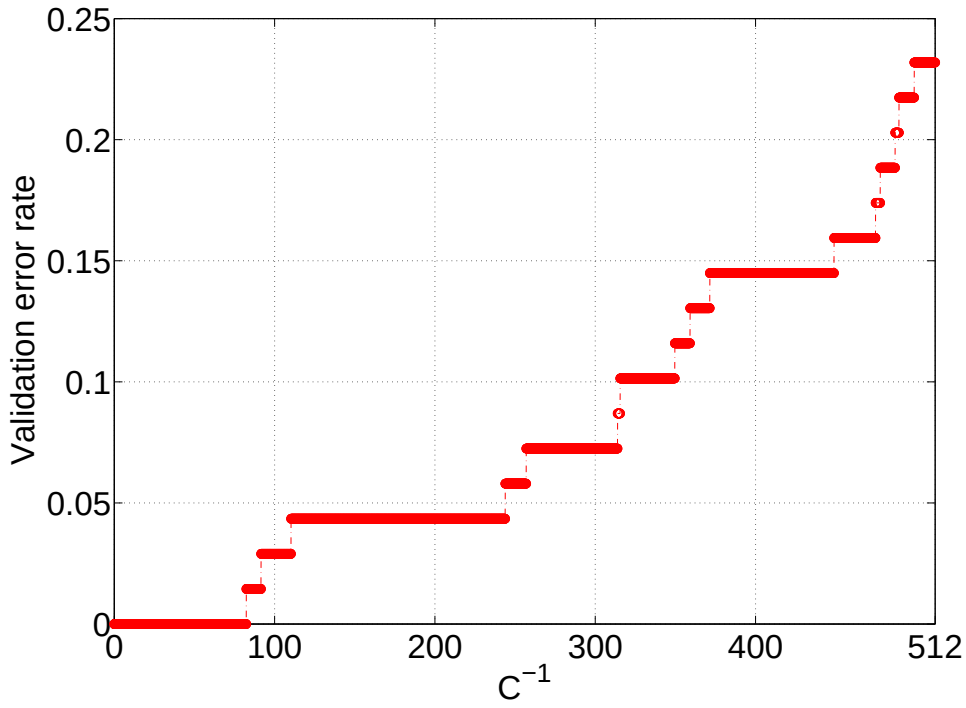


Figure 1.4: Validation error rate for different values of $\lambda = C^{-1}$ for Sonar data set.

data, $\Omega(f)$ is the regularizer reflecting the learning capacity of the predictor and C is the regularizer parameter. The success of SVM depends highly on the regularization parameter C, as it balances the trade-off between the learning capacity of predictor f and the empirical loss [79, 81]. This is consistent with the practical experience that different choices of C result in very different generalization performance of SVM. To illustrate this, Fig 1.4 shows the standard validation error rate of SVM [1] with respect to C^{-1} using Linear kernel on the Sonar data set [1]. It is clear from this figure that the validation error rate can change from 0% to 24 % among the range $C^{-1} \in [2^{-8}, 2^{9}]$.

As mentioned before, the purpose of model selection is to improve the generalization performance, so the procedure of tuning C involves a validation set and an appropriate

[1] This is implemented by the software ISVMP available at: http://guppy.mpe.nus.edu.sg/∼mpeongcj/ongcj.html

validation function. The value of C that optimizes the validation function over the validation set is the optimal C. In the prototypical binary SVM classifier, the validation functions are commonly chosen as the error rate, weighted error rate, percentage of correctly predicted positive examples, or variations thereof. As these validation functions are not smooth functions of C, tuning C in SVM is often resorted to some heuristic or approximated methods, like grid search method or gradient-based method with approximated validation function. These methods will be reviewed in details in Chapter 2.

1.2 Motivations

In this thesis, a wrapper feature selection method for multi-layer perceptron (MLP) neural networks is proposed in Chapter 3 and another wrapper feature selection method for support vector regression (SVR) is proposed in Chapter 4. Then, a filter feature selection method based on mutual information estimation is proposed in Chapter 5. At last, a new model selection method to optimally choose regularization parameter C of SVM is proposed in Chapter 6. The motivations for each of them are provided next.

MLP neural network and SVR are well known learning algorithms and have been successfully used in many applications [5, 6, 20]. To our knowledge, the wrapper feature selection methods for these two algorithms are still limited. One plausible reason is that most existing wrapper methods only focus on binary classification problems while MLP and SVR deal with multi-class classification and regression problems. It is worthy

to note that straightforward adaptation by discretizing (or binning) the continuous output variable into several classes is not always desirable as substantial loss of important ordinal information may result.

Aiming to provide good candidates of wrapper feature selection methods for MLP neural network and SVR, Chapters 3 and 4 propose new feature selection methods using probabilistic outputs of MLP neural networks and SVR, respectively. The results on extensive experiments show the advantage of these two methods over other benchmark methods.

Mutual information based feature selection methods are well known filter feature selection methods. These methods measure the importance of a set of features by evaluating the dependency between this set of features and the output variable, and they often use the forward search strategy. The review of this kind of methods will be provided in Chapter 2. As mentioned before, filter feature selection methods and forward search strategy have limited capability to handle the interacting effect of features.

To alleviate this issue, Chapter 5 proposes a new mutual information based feature selection method. This method is also a filter method but uses a backward search strategy. The experimental results verify the effectiveness of the proposed method on the issue of interacting effect of features.

Proper tuning of regularization parameter C of SVM is important for successful implementation of SVM. However, to the best of our knowledge, there is no existing model selection method that can yield the global optimal C of typical validation functions for

SVM. Most existing methods are approximating the global solution based on grid search strategy or others.

Aiming to resolve this problem, Chapter 6 proposes a new model selection method that guarantees the global optimum of C on a family of common validation functions. This is validated by numerical experiments on large-scale real world data sets.

1.3 Organization

This thesis is arranged as follows:

Chapter 2: This chapter provides reviews of some learning methods to be used in the subsequent chapters. Several relevant filter and wrapper feature selection methods are also reviewed. This chapter ends with a review of some model selection methods especially for hyper parameter tuning of SVM.

Chapter 3: This chapter presents a new wrapper-based feature selection method for MLP neural networks using its probabilistic outputs. This method measures the importance of a feature by the feature's sensitivity with respect to the posterior probability over the whole feature space. This chapter also contains extensive experiments on artificial and real data sets showing the performance comparison between the proposed method and some benchmark methods.

Chapter 4: This chapter presents a new wrapper-based feature selection method for Support Vector Regression (SVR) using its probabilistic predictions. As this feature

ranking criterion is not directly computable, two approximations of this criterion are discussed. This chapter also reports the result of numerical experiment involving the proposed and benchmark methods, tested on artificial and real-world data sets .

Chapter 5: This chapter proposes a new filter-based feature selection method using mutual information. Unlike other mutual information based method, the proposed method measures the importance of a feature in a backward selection framework with the consideration of all features. This chapter also discusses two well-known density estimation methods needed for the computation of the proposed mutual information method. The effectiveness and efficiency of the proposed method are tested with other benchmark methods in numerical experiments.

Chapter 6: This chapter proposes a method to tune the regularized parameter of SVM classifiers. This method can obtain the global optimal C value of the non-smooth validation functions in SVM. The proposed method relies highly on the regularization solution path of SVM over a range of C. The effectiveness of the proposed method evaluated on large scale real-world data sets is also reported in this chapter.

Chapter 7: This chapter concludes this thesis and summarizes its contributions. Directions of future research are also suggested.

Chapter 2

Review

This chapter reviews learning methods used in the later chapters and existing feature selection methods and model selection methods in the literature. For convenience, notations frequently used in this thesis are first introduced. Let $\mathbb{R}$ be the set of real numbers. Data set $\mathscr{D} = \{x_i, y_i\}$, $i \in \mathscr{I}_\mathscr{D} := \{1, \cdots, N\}$ is assumed given with $x_i \in \mathbb{R}^d$ being the i^{th} sample having d features; $\mathscr{I} = \{1, \cdots, d\}$ is the set of indices of all features in $\mathscr{D}$; y_i is the label or output of sample x_i and it can take value $y_i \in \{-1, +1\}$ for binary classification problems, $y_i \in \{1, \cdots, c\}$ for c-class classification problems or $y_i \in \mathbb{R}$ for regression problems. If $\mathscr{S}, \mathscr{Q}$ are two sets, $|\mathscr{S}|$ refers to its cardinality and $\mathscr{S} \backslash \mathscr{Q} := \{x | x \in \mathscr{S}, x \notin \mathscr{Q}\}$ the set difference. Also, $|\mathscr{D}| = |\mathscr{I}_\mathscr{D}|$. Furthermore, $x_i^j \in \mathbb{R}$ is the value of the j^{th} feature of the i^{th} sample in $\mathscr{D}$; the double subscripted symbol $x_{-j,i} \in \mathbb{R}^{d-1}$ refers to the i^{th} sample after the j^{th} feature has been removed from x_i. Equivalently, $x_{-j,i} = Z_j^d x_i$ where Z_j^d is the $(d-1) \times d$ matrix obtained by removing the j^{th} row of the $d \times d$ identity matrix. If r is a random variable, $p(r)$, $\hat{p}(r)$,$P(r)$ and $\mathbb{E}_r$

refer to its density function, estimate of its density function, probability and expectation respectively.

2.1 Learning Methods

2.1.1 Support Vector Machine

The formulations of Support Vector Machine (SVM) and Support Vector Regression (SVR) [81] are provided in this and next subsections. As their applications on classification and regression problems are well known, limited commentary are provided.

SVM is a classification tool of finding the maximum margin hyperplane to separate two classes. The standard two-class SVM primal problem (SVM-PP) with hinge loss $L(\zeta) = \max(0, \zeta)$ is given by:

$$\min_{w,b,\zeta} \ \frac{1}{2} w'w + C \sum_{i \in \mathscr{I}_{\mathscr{D}}} \zeta_i \tag{2.1}$$

$$y_i(w'\phi(x_i) + b) \geq 1 - \zeta_i, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \tag{2.2}$$

$$\zeta_i \geq 0, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \tag{2.3}$$

where $C > 0$ is the regularization parameter, $\phi(x_i)$ is a vector in the high dimensional Hilbert space, $\mathscr{H}$, mapped into by the function $\phi : \mathbb{R}^d \to \mathscr{H}$, w and b are the normal vector and the bias of the separating hyperplane $H := \{\phi(x) | w'\phi(x) + b = 0\}$ respectively. To allow misclassified samples, the non-negative slack variables ζ's are introduced to

enforce inequality constraints (2.2).

In the objective function (2.1), $\frac{1}{2}\, w'w$ is the inverse of the margin between the data in classes $+1$ and -1, and the hinge loss term $\sum_{i\in\mathscr{I}_\mathscr{D}} \zeta_i$ ($\zeta_i \geq 0$) characterizes the degree of misclassification of all samples in $\mathscr{D}$. The former corresponds to the regularizer $\Omega(f)$ in the regularized empirical risk minimization form (1.1) in subsection 1.1.2, while the latter corresponds to the empirical loss $R_{emp}(f)$.

In practice, SVM-PP is often solved by its dual problem (SVM-DP). By introducing Lagrange multiplier α_i for each inequality in (2.2) and γ_i for (2.3), the Lagrange primal function is constructed as

$$L_p: \quad \frac{1}{2}w'w + C\sum_i \zeta_i - \sum_i \alpha_i[y_i(w'\phi(x_i)+b) - 1 + \zeta_i] - \sum_i \gamma_i\zeta_i. \tag{2.4}$$

Setting its derivatives to zero, this gives

$$\begin{aligned}
\frac{\partial}{\partial w}: &\quad w = \sum_i \alpha_i y_i \phi(x_i) \\
\frac{\partial}{\partial b}: &\quad \sum_i \alpha_i y_i = 0 \\
\frac{\partial}{\partial \zeta_i}: &\quad C - \alpha_i = \gamma_i.
\end{aligned}$$

And the Karush-Kuhn-Tucker (KKT) conditions:

$$\begin{aligned}
\alpha_i[y_i(w'\phi(x_i)+b) - 1 + \zeta_i] &= 0 \\
\gamma_i\zeta_i &= 0.
\end{aligned}$$

Putting the above expressions in (2.4), SVM-DP is given by

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i\in\mathscr{I}_{\mathscr{D}}}\sum_{j\in\mathscr{I}_{\mathscr{D}}}\alpha_i\alpha_j y_i y_j K(x_i,x_j) - \sum_{i\in\mathscr{I}_{\mathscr{D}}}\alpha_i \tag{2.5}$$

$$0 \le \alpha_i \le C, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \tag{2.6}$$

$$\sum_i \alpha_i y_i = 0 \tag{2.7}$$

where $K(x_i,x_j) = \phi(x_i)'\phi(x_j)$. The continuous output function of SVM is

$$f(x) = \sum_{i\in\mathscr{I}_{\mathscr{D}}}\alpha_i y_i K(x_i,x) + b. \tag{2.8}$$

where α_i refers to the optimal solution obtained from solving SVM-DP. The decision function is

$$\tilde{y}(x) = \text{sign}(f(x)). \tag{2.9}$$

2.1.2 Support Vector Regression

Similar to SVM, standard SVR [81, 73] with hinge loss $L(\zeta) = \max(0,\zeta)$ is also under the framework of regularized empirical risk minimization (1.1). More exactly, the SVR

Primal Problem (SVR-PP) over w, b, ζ, ζ^* is given by:

$$\min_{w,b,\zeta,\zeta^*} \quad \frac{1}{2}w'w + C\sum_{i\in\mathscr{I}_\mathscr{D}}(\zeta_i + \zeta_i^*) \tag{2.10}$$

$$s.t. \quad y_i - w'\phi(x_i) - b \leq \varepsilon + \zeta_i, \quad \forall i \in \mathscr{I}_\mathscr{D} \tag{2.11}$$

$$w'\phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*, \quad \forall i \in \mathscr{I}_\mathscr{D} \tag{2.12}$$

$$\zeta_i, \zeta_i^* \geq 0, \qquad \forall i \in \mathscr{I}_\mathscr{D}. \tag{2.13}$$

where x is mapped into a high dimensional Hilbert space, $\mathscr{H}$, by the function $\phi : \mathbb{R}^d \rightarrow \mathscr{H}$, and $w \in \mathscr{H}$, $b \in \mathbb{R}$ are variables that define $f(x)$. ζ_i, ζ_i^* are the non-negative slack variables needed for enforcing constraints (2.11) and (2.12). The regularization parameter, $C > 0$, tradeoffs the size of w and the amount of slack variables while parameter, $\varepsilon > 0$, specifies the allowable deviation of the $f(x_i)$ from y_i. In practice, SVR-PP is often solved through its Dual Problem (SVR-DP):

$$\max_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*} \quad -\frac{1}{2}\sum_{i\in\mathscr{I}_\mathscr{D}}\sum_{j\in\mathscr{I}_\mathscr{D}}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i,x_j) - \varepsilon\sum_{i\in\mathscr{I}_\mathscr{D}}(\alpha_i + \alpha_i^*) + \sum_{i\in\mathscr{I}_\mathscr{D}} y_i(\alpha_i - \alpha_i^*) \tag{2.14a}$$

$$s.t. \quad \sum_{i\in\mathscr{I}_\mathscr{D}}(\alpha_i - \alpha_i^*) = 0, \quad 0 \leq \alpha_i \leq C, \;\; 0 \leq \alpha_i^* \leq C, \quad i \in \mathscr{I}_\mathscr{D} \tag{2.14b}$$

where α_i and α_i^* are the respective Lagrange multipliers of (2.11) and (2.12),

$$w = \sum_{i\in\mathscr{I}_\mathscr{D}}(\alpha_i - \alpha_i^*)\phi(x_i) \tag{2.15}$$

and $K(x_i, x_j) = \phi(x_i)'\phi(x_j)$. Using these expressions, the regressor function of SVR is known to be

$$f(x) = w'\phi(x_i) + b = \sum_{i \in \mathscr{I}_{\mathscr{D}}} (\alpha_i - \alpha_i^*) K(x_i, x) + b. \tag{2.16}$$

2.1.3 Entropy and Mutual Information

Entropy of a random variable is a measure of its associated uncertainty while mutual information of two random variables is the reduction in uncertainty of one variable given knowledge of the other. In this sense, mutual information also measures the amount of dependency between the two variables.

Let r, q and t be any three random variables. The entropy, joint entropy and conditional entropy are respectively [17]

$$H(r) = -\int p(r) \log p(r) dr = \mathbb{E}_r \left[-\log p(r) \right] \tag{2.17}$$

$$H(r,q) = -\int\int p(r,q) \log p(r,q) dr dq = \mathbb{E}_{r,q} \left[-\log p(r,q) \right] \tag{2.18}$$

$$H(q|r) = -\int\int p(r,q) \log p(q|r) dr dq = \mathbb{E}_{r,q} \left[-\log p(q|r) \right]. \tag{2.19}$$

The dependency between r and q can be measured by their mutual information:

$$I(r;q) = \int\int p(r,q) \log \frac{p(r,q)}{p(r)p(q)} dr dq = \mathbb{E}_{r,q} \left[\log \frac{p(r,q)}{p(r)p(q)} \right] \tag{2.20}$$

From (2.17)-(2.20), it is easy to show that

$$I(r;q) = H(r) - H(r|q) = H(q) - H(q|r) = H(r) + H(q) - H(r,q). \tag{2.21}$$

By generalizing the concepts of entropy and mutual information, conditional mutual information, e.g. the mutual information between r and q given t, is given by

$$\begin{aligned} I(r;q|t) &= \mathbb{E}_{r,q,t}\left[\log \frac{p(r,q|t)}{p(r|t)p(q|t)}\right] \\ &= \mathbb{E}_{r,q,t}\left[\log \frac{p(r,q)}{p(r)p(q)} + \log \frac{p(r,q|t)p(q)}{p(r,q)p(q|t)} + \log \frac{p(r)}{p(r|t)}\right] \\ &= I(q;r) - I(t;r) + I(t;r|q). \end{aligned} \tag{2.22}$$

It measures the dependency between r and q given the knowledge of variable t.

Using appropriate combinations of joint and marginal density functions, mutual information can provide relationship among random variables that are beyond that of first and second-order statistics [4, 17, 49, 13, 21, 24]. For this reason, they have been used in feature selection methods [4, 47, 53, 21, 23, 83, 46] in the literature. These are reviewed in the later part of this chapter.

2.1.4 Bounds of Generalization Performance

As mentioned in Chapter 1, the goodness of a learning algorithm is often evaluated by its generalization performance — the performance of the learning algorithm on unseen data. In practice, the unseen data is often in the form of a separate data set or as one

fold in an n-fold cross-validation process or just one sample in a Leave-One-Out (LOO) procedure. In the later part of this chapter, we will review that generalization performance, especially LOO generalization performance, has often been used as the criterion for feature selection and model selection. However, implementation of LOO procedure is quite computationally expensive, as a learning algorithm has to be trained and tested for N times if data set $\mathscr{D}$ is given. Moreover, LOO generalization performance is often nondifferentiable with respect to the interested parameters.

To alleviate these issues, some bounds of LOO generalization performance for learning algorithms are given. For example, radius margin bound and span bound for SVM (2.1), without considering loss $L(\zeta)$ and bias b, are firstly proposed by Vapnik [81] and Vapnik and Chapelle [80] respectively. Specifically, with the same meanings of w, α and K in subsection 2.1.1, the radius margin bound is

$$4R^2||w||^2 \tag{2.23}$$

where R is the radius of the smallest sphere containing all the points $\phi(x_i),\ \forall i \in \mathscr{I}_{\mathscr{D}}$ and it can be computed by solving the following optimization problem:

$$\begin{aligned} R^2 = \max_{\boldsymbol{\gamma}}\ & 1 - \boldsymbol{\gamma}' K \boldsymbol{\gamma} \\ & \text{s.t. } \gamma_i \geq 0,\ \sum_i \gamma_i = 1,\ i \in \mathscr{I}_{\mathscr{D}}. \end{aligned} \tag{2.24}$$

The span bound is

$$\sum_{i \in \mathscr{I}_D} \alpha_i \mathbb{S}_i^2 \tag{2.25}$$

where $\mathbb{S}_i$ is the distance between the point $\phi(x_i)$ and the following set

$$\Gamma_i = \left\{ \sum_{j \neq i, \alpha_j > 0} \gamma_j \phi(x_j) \mid \sum_{j \neq i} \gamma_j = 1 \right\}. \tag{2.26}$$

Note the assumption that the set of support vectors remains the same in LOO procedure is needed in span bound. The continuity and differentiability of these bounds are investigated in [12]. Later, the improvement of these bounds and the extension of them to other forms of SVM are addressed in Chung and Lin [15]. Motivated by these preliminary work on SVM problem, Chang et al. [11] further propose radius margin bound and span bound for SVR problem.

2.2 Feature Selection Methods

In this section, several related existing feature selection methods are reviewed and they serve as benchmarks to the proposed methods in numerical experiments of Chapters 3, 4 and 5.

2.2.1 Filter Methods

Fisher Score Method

Fisher score [31] is probably the easiest and most widely-used filter method for classification problems. It is the ratio of "between variance" and "within variance" of each feature. In a c-class $\{\omega_1, \cdots, \omega_c\}$ classification problem, the Fisher score for the j^{th} feature is defined as

$$S^{Fscore}(j) = \frac{\sum_{k=1}^{c} N_k(\mu_k^j - \mu^j)^2}{\sum_{k=1}^{c} \sum_{x_i \in \omega_k} (x_i^j - \mu_k^j)^2}, \quad \forall j \in \mathscr{I} \tag{2.27}$$

where N_k is the number of samples belonging to class ω_k, $\mu_k^j = \frac{1}{N_k} \sum_{x_i \in \omega_k} x_i^j$ is the mean of j^{th} feature in the k^{th} class and $\mu^j = \sum_{k=1}^{c} N_k \mu_k^j / N$ is the mean of μ_k^j over all the classes. With these notations, $(\mu_k^j - \mu^j)^2$ in the numerator of (2.27) amounts to the discrepancy between the centroid of class j and the centroid of all classes and such discrepancy is weighted by N_k, while $\sum_{x_i \in \omega_k} (x_i^j - \mu_k^j)^2$ in the denominator amounts to the variance within class j. The intuitive meaning of this method is that the important feature should have better discrimant ability (i.e. larger "between variance" and smaller "within variance"). Therefore, the greater the score of (2.27) the greater the feature's importance.

The underlying assumption of Fisher score method is that features are assumed independent and they are ranked according to their own estimated individual predictive capabilities. This assumption also exists in other naive filter methods including Kolmogorov-

Smirnov test [32] or Pearson correlation [56].

Mutual Information Based Methods

In the past decades, various mutual information based feature selection methods are proposed for classification and regression problems [4, 47, 53, 21, 23, 83]. These methods are often used in a forward selection framework. The forward selection framework is implemented in an iterative procedure whereby, in each iteration, the most important feature in $\mathscr{D}$ is identified among a set of remaining features based on some criterion. This most important feature is then removed from the set of remaining features and added to a set of identified features. Several criteria have been proposed under this framework. Suppose $z \in \mathbb{R}^v$ is a vector obtained by taking v ($v < d$) of the d features from $x \in \mathbb{R}^d$. The most direct criterion is to find the most appropriate z vector that maximizes the mutual information $I(z;y)$. This is reasonable since the aim is to reduce the uncertainty of y given the information of z. Such a criterion can easily be incorporated in a forward selection framework. Battiti [4] and Kwak et al. [46] propose the use of

$$I(z_{+j};y) \tag{2.28}$$

for feature $j \in \mathscr{I} \setminus \mathscr{I}_{\ell-1}$ at the ℓ iteration. Here, $z_{+j} \in \mathbb{R}^\ell$ is the augmented vector of z with an additional feature j or, equivalently, is derived from the vector x with features from $\mathscr{I}_{\ell-1} \cup \{j\}$, $\mathscr{I}$ being the set of all features, $\mathscr{I}_{\ell-1}$ the set of identified features till iteration ℓ and $\mathscr{I} \setminus \mathscr{I}_{\ell-1}$ the set of remaining features at the ℓ iteration. From the definition of mutual informatin, the computation of (2.28) requires knowledge of $p(z_{+j})$,

$p(z_{+j},y)$ and $p(y)$ at every ℓ. As these functions are typically not available, estimations are needed. To facilitate these estimations, several related criteria have been proposed. In this direction, Battiti [4] proposes

$$I(x^j;y) - \frac{1}{\beta} \sum_{i \in \mathscr{I}_{\ell-1}} I(x^j;x^i), \tag{2.29}$$

where β is a user-determined weighting parameter. As a result, the evaluation of (2.29) requires only estimations of low-dimensional density functions, and is therefore computationally amenable. The criterion (2.29) also has a slightly different meaning from (2.28). Since $\sum_{i \in \mathscr{I}_{\ell-1}} I(x^j;x^i)$ is the sum of measures of dependence of x^j and x^i for all $i \in \mathscr{I}_{\ell-1}$, criterion (2.29) captures the additional dependency between x^j and y that is not present in $\sum_{i \in \mathscr{I}_{\ell-1}} I(x^j;x^i)$. Several variants of criterion (2.29) are proposed in the literature [47, 53, 21] by modifying the second term in (2.29). These criteria include

$$I(x^j;y) - \frac{1}{\beta} \sum_{i \in \mathscr{I}_{\ell-1}} \frac{I(y;x^i)}{H(x^i)} I(x^j;x^i), \tag{2.30}$$

$$I(x^j;y) - \frac{1}{|\mathscr{I}_{\ell-1}|} \sum_{i \in \mathscr{I}_{\ell-1}} I(x^j;x^i), \tag{2.31}$$

$$I(x^j;y) - \frac{1}{|\mathscr{I}_{\ell-1}|} \sum_{i \in \mathscr{I}_{\ell-1}} \frac{I(x^j;x^i)}{\min\{H(x^j),H(x^i)\}}. \tag{2.32}$$

In addition, Fleuret [23] and Vasconcelos et al. [83] approximate (2.28) in different ways respectively:

$$\min_{i \in \mathscr{I}_{\ell-1}} I(y;x^j|x^i) = I(x^j;y) + \min_{i \in \mathscr{I}_{\ell-1}} [I(x^j;x^i|y) - I(x^j;x^i)], \tag{2.33}$$

$$\sum_{i \in \mathscr{I}_{\ell-1}} I(y; x^j | x^i) = I(x^j; y) - \sum_{i \in \mathscr{I}_{\ell-1}} [I(x^j; x^i) - I(x^j; x^i | y)]. \tag{2.34}$$

where $I(y, x^j | x^i)$ is the dependence between x^j and y given x^i and the last equalities in both equations follow from the definition of conditional mutual information (2.22).

Criteria (2.29) to (2.34) have been successfully used in some applications due to their simplicity and efficiency, but they can suffer from the following drawbacks. First, while the use of (conditional) mutual information terms with 2 or 3 features simplifies the computation, these criteria may not be effective in capturing effects of 3 or more interacting features. Second, the first step in these forward feature selection methods is crucial as it determines the most important feature. However, all above methods select the most important feature by the criterion of $\arg\max_{j \in \mathscr{J}} I(y; x^j)$, which assumes all features are independent. It is therefore very possible that forward scheme incorrectly chooses the most important feature. Third, in the subsequent steps of forward feature selection, criteria (2.29) to (2.34) again ignore the interacting effect of the incumbent feature with those yet to be identified [32]. The above three drawbacks in existing methods could lead to performance degradation on feature selection. These issues will be further studied in Chapter 6.

Dependence Maximization Method

Recently, Song et al. [74] propose a sophisticated filter method which appears to be quite effective in dealing with data sets having interactive features. This method has the similar idea with the mutual information based feature selection method: the important

features should have the maximum dependence with target variable. They only differ in the way of measuring the dependence of two variables.

A dependence maximization method uses cross-covariance in the kernel space, known as the Hilbert-Schmidt norm of cross-covariance operator (HSIC) [28], as dependence measure between feature variables and target variable. More exactly, suppose (x,y) and $(\tilde{x},\tilde{y})$ are independently drawn from $\mathscr{D}$. Let x and $\tilde{x}$ be mapped into a reproducing kernel Hilbert space (RKHS) $\mathscr{H}$ by $\phi : \mathbb{R}^d \to \mathscr{H}$ and y and $\tilde{y}$ be mapped into another RKHS $\mathscr{G}$ by $\varphi : \mathscr{Y} \to \mathscr{G}$ where $\mathscr{Y} = \mathbb{R}$ in regression problems, $\mathscr{Y} = \{-1,+1\}$ in binary classification problems or $\mathscr{Y} = \{1,\cdots,c\}$ in multi-class classification problems. The HSIC between input variables and target variable is defined as:

$$\begin{aligned}\mathrm{HSIC}(\mathscr{H},\mathscr{G},\mathscr{D}) =& \|\mathbb{E}_{xy}[(\phi(x)-\mathbb{E}_x(\phi(x))\otimes(\varphi(y)-\mathbb{E}_y(\varphi(y)))]\|^2_{\mathrm{HS}} \\ =& \mathbb{E}_{x\tilde{x}y\tilde{y}}[K(x,\tilde{x})L(y,\tilde{y})]+\mathbb{E}_{x\tilde{x}}[K(x,\tilde{x})]\mathbb{E}_{y\tilde{y}}[L(y,\tilde{y})] \\ &-2\mathbb{E}_{xy}[\mathbb{E}_{\tilde{x}}[K(x,\tilde{x})]\mathbb{E}_{\tilde{y}}[L(y,\tilde{y})]]\end{aligned} \tag{2.35}$$

where $\otimes$ is the tensor product, $K(x,\tilde{x}) = \phi(x)'\phi(\tilde{x})$ and $L(y,\tilde{y}) = \varphi(y)'\varphi(\tilde{y})$. As claimed in [28, 74], the expression (2.35) can measure the non-linear dependence between x and y, since both of them are mapped into high dimensional space.

In term of computation, each expectation term in the last equality of (2.35) can be approximately computed by the U-statistics, and an unbiased estimator of (2.35) is given by

$$\mathrm{HSIC}_1(\mathscr{H},\mathscr{G},\mathscr{D}) = \frac{1}{N(N-3)}\left[\mathrm{trace}\tilde{K}\tilde{L}+\frac{\mathbf{1}'\tilde{K}\mathbf{1}\mathbf{1}'\tilde{L}\mathbf{1}}{(N-1)(N-2)}-\frac{2}{N-2}\mathbf{1}'\tilde{K}\tilde{L}\mathbf{1}\right] \tag{2.36}$$

where **1** is the column vector with all elements being 1, and $\tilde{K}$ and $\tilde{L}$ are the same as K and L respectively except that their diagonal entries are all set to zero.

Further exploitation of the dependence measure (2.35) and its computation (2.36) can be found in [28]. In the feature selection method of [74], the dependence measure (2.36) is used in a backward feature selection scheme, i.e., the least-important feature is successively removed at each time.

2.2.2 Wrapper Methods

Maximum Output Information Method

Mutli-layer perceptron (MLP) neural network is a well-known machine learning method. Maximum Output Information (MOI) [72] is a recently proposed wrapper method for MLP, and appears to outperform other existing wrapper methods for MLP, such as neural-network feature selector (NNFS) [70] and artificial neural net input gain measurement approximation (ANNIGMA) [40].

MOI method uses a procedure, called *information back-propagation*, to assign a score to each feature. Herein, the *information* refers to the mutual information between the *true* label y and the *predicted* label $\hat{y}$ obtained from the trained MLP. When this *information* traverses the trained MLP neural network from output layer to input layer, the resultant score to each feature can measure the contribution of the feature w.r.t. the dependency between y and $\hat{y}$. These scores can therefore be used to rank the features.

The idea of this method appears sound and attractive, but the procedure of *information back-propagation* is not directly computable and several heuristics are used for its approximations. The details of the heuristics used can be found in [72].

SVM-RFE Method

Due to the success of support vector regression (SVR), feature selection for SVR has also attracted considerable works in the past few years. This subsection and the next provide the review of a few feature selection methods particularly for SVR.

SVM-RFE, RFE short for Recursive Feature Elimination, is a well-known wrapper-based feature selection method for classification problems with reported good performance [31, 33, 64]. Guyon et al. [33] also suggest that this method is applicable to regression problems. In this case, SVM-RFE measures the importance of a feature by the sensitivity of the cost function of SVR with and without this feature. The importance of the j^{th} feature is evaluated by

$$S^{\Delta\|w\|^2}(j) = \mid \|w\|^2 - \|w_{-j}\|^2 \mid, \ \ \forall, j \in \mathscr{I} \tag{2.37}$$

where w refers to expression (2.15) in subsection 2.1.2 and its variant w_{-j} is obtained from

$$w_{-j} = \sum_{i \in \mathscr{I}_{\mathscr{D}}} (\alpha_i - \alpha_i^*)\phi(x_{-j,i}) \tag{2.38}$$

where α_i and α_i^* $\forall i$ are obtained from the trained SVR on $\mathscr{D}$ and $x_{-j,i}$ is the i-th input sample of data set $\mathscr{D}_{-j} := \{(x_{-j,i}, y_i) | x_{-j,i} = Z_j^d x_i \text{ for all } (x_i, y_i) \in \mathscr{D}\}$. Expression (2.38) implicitly assumes that the support vectors remain unchanged when a feature is removed. Hence, the expensive procedure of retraining SVR with $\mathscr{D}_{-j}$ is avoided.

This method has been successfully used for regression application [29] with notable success.

Leave-One-Out Error Bounds Methods

Based on the preliminary work of leave-one-out bound mentioned in subsection 2.1.4, Rakotomamonjy [65] proposes a few feature ranking criteria using leave-one-out error bounds of SVR as feature importance index. Although the used SVR model in [65] is SVR with the square loss $L(\zeta) = \max(0, \zeta)^2$, but the extension of these criteria to standard SVR model with hinge loss $L(\zeta) = \max(0, \zeta)$ as used in (2.10) is straightforward.

Rakotomamonjy [65] compared his proposed criteria in extensive experiments and concluded that the best two criteria are the radius-margin bound,

$$S^{B1}(j) = R_{-j}^2 \sum_{i \in \mathscr{I}_{\mathscr{D}}} (\alpha_{-j,i} + \alpha_{-j,i}^*), \tag{2.39}$$

and span estimate bound

$$S^{B2}(j) = \sum_{i \in \mathscr{I}_{\mathscr{D}}} (\alpha_{-j,i} + \alpha_{-j,i}^*) \mathbb{S}_{-j,i}^2. \tag{2.40}$$

In (2.39), R_{-j} is the radius of the smallest sphere containing all the points $\phi(x_{-j,i})$, $i \in \mathscr{I}_{\mathscr{D}}$, and $\{\alpha_{-j,i} | i \in \mathscr{I}_{\mathscr{D}}\}$ and $\{\alpha^*_{-j,i} | i \in \mathscr{I}_{\mathscr{D}}\}$ are SVR solution with data set $\mathscr{D}_{-j}$. In (2.40), $\mathbb{S}^2_{-j,i}$ is the squared distance of $\phi(x_{-j,i})$ to the span of all other support vectors $\{\phi(x_{-j,t}) | t \in \mathscr{F} \setminus \{i\}\}$ with $\mathscr{F} = \{t | 0 < \alpha_{-j,t} + \alpha^*_{-j,t} < C\}$. More details of computing R_{-j} and $\mathbb{S}^2_{-j,i}$ can be found in [12, 11].

Criteria (2.39) and (2.40) measure the importance of a feature by its sensitivity to the leave-one-out error bound, and the feature with the smallest error bound is considered as the non-important feature.

2.3 Model Selection Methods

The procedure of tuning the regularization parameter, C, is a well-known problem in the study of Support Vector Machine (SVM) classifier. As mentioned in Chapter 1, one difficulty of tuning C of common validation functions (such as the error rate, weighted or balanced error rate, precision, recall or variations thereof) is that these functions are not smooth functions of C and the determination of the optimal C is not easy.

To solve this problem, the techniques of sampling among C values and approximating validation function are often used in practice. These methods are reviewed next.

2.3.1 Grid Search Method

The grid search method is direct in the sense that it computes validation functions over a set of pre-specified C values and chooses the minimum among them. This method is widely used in practice, including the standard software packages LIBSVM [10], LIBLINEAR [22], SVM^{light} [41] and Weka [88]. It is reported [19] that grid search method can yield comparable or better performance in comparison with some approximated validation function methods discussed in 2.3.2.

Generally, there is no guarantee that grid method can find the global optimal C value. The chance of getting a good approximation to the optimal C increases when the grid gets dense. However, the corresponding computational cost also increases.

2.3.2 Gradient-based Methods

Some methods find the optimal C by approximating common validation functions [11, 15, 12, 45]. Among them, Chapelle et al. [12] suggest several measures for such a purpose. These include various bounds on the generalization error like the radius margin bound and span bound mentioned in subsection 2.1.4. Empirical evaluations of several measures have also been reported [19]. Another popular choice is the sigmoidal approximation of the output function of SVM [45]. All these approximations are used for the procedure of tuning C as they are smooth functions of C and can facilitate numerical determination of optimal C via standard gradient-based optimization algorithms.

However, as approximations, the connection of these smooth functions to the true validation function is not direct. They are also known to have multiple local stationary points, making the determination of the global optimum difficult for gradient-based algorithms.

2.3.3 Regularization Solution Path of SVM

The above two model selection methods do not guarantee the optimal C value of typical validation functions. Chapter 6 of this thesis proposes a method that does. It is based on the availability of complete solution path of SVM on a wide range of C. This solution path approach is now reviewed.

Hastie et al. [34] first propose an approach (hereafter referred to as the *SVMpath*) on providing SVM solutions for a wide range of values of the regularization parameter, C. It is based on a one-dimensional tracking of the Karush-Kuhn-Tucker (KKT) optimality condition of the dual problem as C changes, resulting in numerical solutions for all values of C. Extensions of *SVMpath* to other problems have also appeared [69, 30, 90], including those for regression problem [85, 86]. Recetnly, Ong et al. [58] present a method, called ISVMP, to improve on the reliability of *SVMpath* so that it can deal with data set having duplicate data points, nearly duplicate points, or points that are linearly dependent in the kernel space.

Apparently, *SVMpath* or ISVMP can facilitate the procedure of tuning C in multiple ways. The most direct way is to replace the SVM solver required in the existing grid search and gradient based methods, with the results of *SVMpath* or ISVMP. However,

such an approach does not avoid the problems of multiple local minimums or the non-smooth routine. A new method using ISVMP is proposed in Chapter 6 that guarantees the global optimum of C on a family of common validation functions.

Chapter 3

Feature Selection via Sensitivity Analysis of MLP Probabilistic Outputs

This chapter proposes a new wrapper-based feature selection method for MLP and is an extension of the earlier work for SVM [71]. This extension is motivated by the popularity of MLP as a classifier/regressor for many pattern recognition problems. Consider the case where the output of the MLP takes the form of $P(\omega_k|x)$, the posterior probability of sample x belonging to class ω_k, for all x in the feature space. The proposed feature selection method, termed Feature-based Sensitivity of Posterior Probabilities (FSPP), uses the sensitivity of $P(\omega_k|x)$ with respect to a feature as the ranking criterion to measure the importance of that feature. In loose terms, this criterion is the aggregate value, over the *entire* feature space, of the absolute difference of $P(\omega_k|x)$ over all classes of k with and without a given feature. As its original form is not easily computable, an approximation is proposed. This approximation, used in an overall feature selection scheme, is

then tested on various artificial and real-world data sets, in comparison to several existing feature selection methods in the literature for MLP. The results show the proposed method performs generally better than the existing methods considered.

The remainder of this chapter is organized as follows. Section 3.1 provides the standard basis of probabilistic MLP neural networks. Section 3.2 gives the detailed account of the proposed feature ranking criterion and its approximation. Section 3.3 outlines the use of the proposed criterion in an overall feature selection scheme. Section 3.4 reports extensive numerical studies of the proposed method in comparison to some existing methods in the literature, followed by the summary given in Section 3.5.

3.1 Preliminary

The structure of the MLP neural network considered in this thesis is shown in Figure 3.1. Note that the neural network with multiple layers can be straightforwardly extended. It is a popular choice for probabilistic neural network [43] and consists of a single-layer hidden neurons with smooth activation functions, an output layer with linear neuron (neuron with linear activation function) and a softmax function after the output neurons. The choice of the smooth activation function used in this thesis is the hyperbolic tangent but other choices may also be used. One hidden layer is used because it is known to have sufficient approximating power [18], [38]. The exact number of the hidden neurons, m, is a hyper-parameter and its value is determined using n-fold cross validation. Let variables b^0, b^1 represent the biases of the input to the respective layers, and W_{ij}^{ℓ} denote

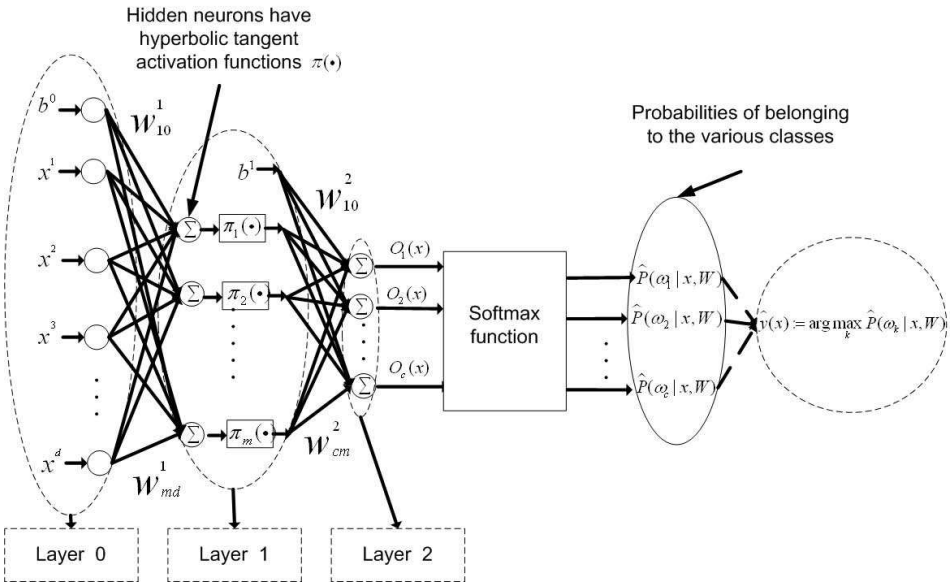


Figure 3.1: Architecture of softmax-based probabilistic MLP.

the values of the weights from the j^{th} neuron of layer $\ell-1$ to the i^{th} neuron of layer ℓ, and also W be the collection of W_{ij}^{ℓ}, $\forall i, j, \ell$, of the network. Then, the output function $O_k(x;W)$ with $k = 1, \cdots, c$ is

$$O_k(x;W) = \sum_{u=1}^{m} W_{ku}^{2} \cdot \pi_u(\sum_{j=1}^{d} W_{uj}^{1} \cdot x^j) \tag{3.1}$$

where $\pi_u(\cdot) = tanh(\cdot)$ is the activation function of u-th neuron in layer 1. The softmax function provides probabilistic estimate from the $O_k(x;W)$ for all $x \in \mathbb{R}^d$ in the form of

$$\hat{P}(\omega_k|x;W) := \frac{e^{O_k(x;W)}}{e^{O_1(x;W)} + e^{O_2(x;W)} + \cdots + e^{O_c(x;W)}}, \quad k = 1, \cdots, c \tag{3.2}$$

where $e^{(\cdot)}$ is the exponential function and $\hat{P}(\omega_k|x;W)$ is the posterior probability of x belonging to ω_k for a given set of W. The determination of W is achieved using the well-established back-propagation update rule for the minimization of the entropy cost

function,

$$E(W) = \sum_{i=1}^{N} \sum_{k=1}^{c} [-\delta_k(x_i) \ln \hat{P}(\omega_k|x_i;W)], \tag{3.3}$$

where $\delta_k(\cdot)$ is the indicator function: $\delta_k(x_i) = 1$ if $y_i = k$ and $\delta_k(x_i) = 0$ otherwise. This cost function has a well-known interpretation: minimizing $E(W)$ corresponds to maximizing the likelihood function of observing the data set $\mathscr{D}$. Suppose W^* is the solution to (3.3), then the predicted label for any $x \in \mathbb{R}^d$ is given by the decision rule:

$$\hat{y}(x) := \arg\max_k \hat{P}(\omega_k|x;W^*). \tag{3.4}$$

3.2 The Proposed Wrapper-based Feature Ranking Criterion for Classification

In c-class classification, the proposed feature-ranking criterion for the j^{th} feature is:

$$S^P(j) = \sum_{k=1}^{c} \int_{\mathbb{R}^d} \mid P(\omega_k|x) - P(\omega_k|x_{-j}) \mid p(x)dx, \tag{3.5}$$

where $x_{-j} \in \mathbb{R}^{d-1}$ is the sample derived from x with the j^{th} feature removed (or equivalently, $x_{-j} = Z_j^d x$), $p(x)$ is the probability density function of x and the integration is taken over the entire feature space. The motivation of above criterion is clear: the greater the absolute difference between $P(\omega_k|x)$ and $P(\omega_k|x_{-j})$ over the feature space, the more important is the j^{th} feature. Clearly, it is a sensitivity of the posterior prob-

abilities with respect to a feature and is hence termed the Feature-based Sensitivity of Posterior Probabilities (FSPP).

The value of $P(\omega_k|x_{-j})$ in (3.5) corresponds to the probabilistic output of softmax-based MLP trained using data $\mathscr{D}_{-j} := \{(x_{-j,i}, y_i)|x_{-j,i} = Z_j^d x_i \text{ for all } (x_i, y_i) \in \mathscr{D}\}$. As x has d features, evaluation of $S^P(j), j = 1, 2, \cdots, d$ requires that retraining of the MLP is performed d times, each time with the data set $\mathscr{D}_{-j}$ for a different j. This is obviously a computationally expensive process. Following the work in SVM by Shen et al. [71], a random permutation (RP) process [8, 59] is used to approximate $P(\omega_k|x_{-j})$ such that the retraining of MLP is avoided. The basic idea of RP process is to randomly permute the values of the j^{th} feature in $\mathscr{D}$ while keeping the values of all other features unchanged. Specifically, let $\{\eta_1, \cdots, \eta_{N-1}\}$ be a set of uniformly distributed random numbers in the interval $(0, 1)$ and $\lfloor \eta \rfloor$ be the largest integer that is less than η. Then, for each i starting from 1 to $N-1$, compute $k = \lfloor N \times \eta_i \rfloor + 1$ and swap the values of x_i^j and x_k^j.

Let $x_{(j)} \in \mathbb{R}^d$ denote the sample derived from x after the values of the j^{th} feature randomly permuted by the RP process and $\mathscr{D}_{(j)} := \{x_{(j),i}, y_i\}_{i=1}^N$ denote the resultant data set. The next theorem states a result on $P(\omega_k|x_{(j)})$ following the RP process and serves as the theoretical basis for the proposed approximation of (3.5).

Theorem 3.2.1.

$$P(\omega_k|x_{(j)}) = P(\omega_k|x_{-j}) \tag{3.6}$$

The proof of this theorem is given Appendix A.

Remark 3.2.1. *As shown in the proof, the result $P(\omega_k|x_{(j)}) = P(\omega_k|x_{-j})$ is validated only needs the process which can destroy the dependence between (ω_k, x_{-j}) and x^j as well as the dependence between x_{-j} and x^j. Therefore, theoretically any procedure with such function can lead to result.*

Random permutation procedure is a good example of this process, especially for the data set with large number of samples. Nevertheless, in practice data set often has limited number of samples, e.g., gene dataset has quite few number of samples. In this case, random permutation may not fully destroy the features dependence. To resolve this problem, multiple times of random permutation might be needed.

The theorem is stated for the case where $P(\omega_k|x), P(\omega_k|x_{(j)})$ and $P(\omega_k|x_{-j})$ are known. In the case where they are approximated from the data set, the equality of (3.6) becomes an approximation. Nevertheless, our numerical experiment shows that the approximation is very good, even when the data is sparse.

Theorem 3.2.1 shows that random permutation of the values of a feature has the same effect as removing the contribution of that feature for classification. Using this fact, (3.5) can be equivalently stated as

$$S^P(j) = \sum_{k=1}^{c} \int_{\mathbb{R}^d} \mid P(\omega_k|x) - P(\omega_k|x_{(j)}) \mid p(x)dx. \tag{3.7}$$

As its true value is not known, $P(\omega_k|x)$ is approximated by $\hat{P}(\omega_k|x) := \hat{P}(\omega_k|x;W^*)$ as in (3.2), obtained from the softmax-based MLP trained using $\mathscr{D}$. Similarly, $P(\omega_k|x_{(j)})$ is approximated by $\hat{P}(\omega_k|x_{(j)})$ obtained using the *same* MLP classifier. Further approx-

imation of the integration over x in (3.7) yields

$$\hat{S}^P(j) = \frac{1}{N}\sum_{k=1}^{c}\sum_{i=1}^{N} \mid \hat{P}(\omega_k|x_i) - \hat{P}(\omega_k|x_{(j),i}) \mid . \tag{3.8}$$

Using (3.8) and the RP process, $\hat{S}^P(j)$ can be computed for $j = 1,\cdots,d$ after a one-time training of the softmax-based MLP classifier, and d times forward computing MLP each time with $\mathscr{D}_{(j)}$ as input. The computational cost of one-time training of MLP has a known complexity [55] of about $O(2\tau N|W|)$, where $|W|$ is the total number of weights in the MLP and τ is the number of learning iterations of MLP training. Suppose the optimal W^* has been obtained. The computational cost of evaluating (3.8) using $\mathscr{D}_{(j)}$ for all $j = 1,\cdots,d$ is about $O(N|W|)$. Hence, the total computational cost is $O((2\tau + 1)N|W|)$. Clearly, this is much cheaper than to retrain the MLP d times which has a cost of $O(2d\tau N|W|)$.

3.3 Feature Selection Scheme

Like other criteria, $\hat{S}^P$ of (3.8) can be used in several ways. It can provide a ranked list of features based on a one-time training of the MLP. It can also be used in more extensive ranking schemes like the well-known recursive feature elimination (RFE) approach [33]. The RFE approach removes the least important feature, as determined by $\hat{S}^P$, recursively from successive training of the MLP. Accordingly, the overall scheme is referred to as MLP-FSPP-RFE and its main steps are listed in Algorithm 1. It has its inputs data set $\mathscr{D}$ and the index set $\mathscr{I} = \{1,\cdots,d\}$. The output is a ranked list of features in the form

of an index set $\mathscr{I}^f = \{i_1^f, \cdots, i_d^f\}$ where $i_j^f \in \mathscr{I}$ for each $j = 1, \cdots, d$ and i_1^f being the index of the most important feature and i_d^f the least.

Algorithm 1: Main steps of MLP-FSPP-RFE feature selection scheme.

Input: $\mathscr{D}$, $\mathscr{I}$
Output: $\mathscr{I}^f := \{i_1^f, \cdots, i_d^f\}$
1 **while** $|\mathscr{I}| > 0$ **do**
2 Let $\ell = |\mathscr{I}|$;
3 **if** $\ell > 1$ **then**
4 Train the softmax-based MLP with $\mathscr{D}$;
5 For each $j \in \mathscr{I}$, compute $\hat{S}^P(j)$ using (3.8);
6 Obtained a ranked list $\mathscr{J} = \{j_1, \cdots, j_\ell\}$, $j_k \in \mathscr{I}$ from $\{\hat{S}^P(j)\}_{j=1}^{\ell}$ such that $\hat{S}^P(j_k) \geq \hat{S}^P(j_{k+1})$ for $k = 1, \cdots, \ell - 1$;
7 Let $i_\ell^f = j_\ell$;
8 Let $\mathscr{I} = \mathscr{I} \setminus j_\ell$ and $\mathscr{D} = \mathscr{D} \setminus \{x_i^{j_\ell} : i \in \mathscr{I}_{\mathscr{D}}\}$;
9 **else**
10 Let $i_1^f = j_\ell$ and $\mathscr{I} = \mathscr{I} \setminus j_\ell$;
11 **end**
12 **end**

With reference to Algorithm 1, the while loop is invoked $d - 1$ times. Each time, the softmax-based MLP is trained with a reduced data set $\mathscr{D}$ (step 4) and produces a ranked list $\mathscr{J}$ of all features in $\mathscr{D}$ (step 6) based on the scores of $\hat{S}^P$. The least important feature (the last element of $\mathscr{J}$) is removed from $\mathscr{I}$ and stored in the ranked list $\mathscr{I}^f$. The corresponding feature is also removed from the data set $\mathscr{D}$ (step 8). The while loop is then invoked on the reduced sets of $\mathscr{I}$ and $\mathscr{D}$ again. This process continues, each time removing the least important feature from $\mathscr{I}$ and storing in the last position of $\mathscr{I}^f$, until $\mathscr{I}$ has only one feature, which becomes the most important feature naturally.

It is worth noting that more than one feature can be removed at one time with a slight modification to step 7 and 8 in the Algorithm 1. Like other wrapper methods, the current

scheme does not involve the re-tuning of the number of hidden neurons in step 4 in the while loop of Algorithm 1. Re-tuning is possible albeit with much higher costs.

3.4 Numerical Experiment

Extensive experiments on both artificial and real-world data sets are conducted to evaluate the performance of the proposed method and three existing MLP feature selection methods mentioned in Section 2.2, Fisher Score (FisherS) [31] of (2.27), Mutual Information (MutualI) [53] of (2.31) and Maximum Output Information (MOI) [72]. Following the procedure of Rätsch [67], the result of the experiment is reported over 30 realizations for all data sets. The subset $\mathscr{D}_{trn}$ is normalized to zero mean and unit standard deviation and its normalization parameters are then used to normalize $\mathscr{D}_{tst}$. $\mathscr{D}_{trn}$ is used for training the softmax-based MLP, including the determination of m, via a 5-fold cross-validation over the grid $[1,2,\cdots,3d]$ for all problems, except for the problems of HillValey and Musk where the grid $[1,2,\cdots,6]$ is used. The grid size is chosen according to the rule-of-thumb that the total number of weights in MLP should be less than the number of training samples. The subset $\mathscr{D}_{tst}$ is used for obtaining an unbiased evaluation of the effectiveness of the underlying feature selection methods. For the case of the MOI method, a separate validation data set is needed for the *information back-propagation* evaluation. Hence, $\mathscr{D}_{trn}$ is further divided into two equal parts: one as $\mathscr{D}_{trn}$ for the training the MLP and the other as $\mathscr{D}_{val}$ for conducting information back-propagation. $|\mathscr{D}_{trn}|$ and $|\mathscr{D}_{tst}|$ are the number of training samples and the number of test samples, respectively.

The presentation of the results follows that by Rakotomamonjy [64] where the (average) test error rates varying with the number of top-ranked features for each method are plotted. The plots are the mean over all realizations of each data set. In each figure, the results of MLP-FSPP-RFE and the existing benchmark methods, FisherS, MutualI, MOI are reported. In addition, for statistical comparison of the methods, paired t-test between the proposed method and each of benchmark methods is conducted on all data sets. Specifically, the null hypothesis is that the mean test errors of the two methods are same and the paired t-test is conducted for a given number of top-ranked features. The p-value obtained in the paired t-test is given and the symbols "$+$" and "$-$" are used to indicate win or loss of the proposed method over that method.

The numerical algorithm for the training of the MLP in our experiments is done using the Netlab package [57], where a scaled conjugate gradient method is used in the optimization of the cost function (3.3).

3.4.1 Artificial Data Sets

Weston's Nonlinear Synthetic Data Sets

This artificial data set has 10 features and 10,000 samples. It is generated according to the procedure in [87]. Only the first two features 1 and 2 are relevant while others are random noise, each taken from a normal distribution, $\mathcal{N}(0,20)$. The target $y \in \{1,2\}$ and the number of samples with $y = 1$ is equal to that with $y = 2$. If $y = 1$, (x^1, x^2) are drawn from two normal distributions $\mathcal{N}(\mu_1, \Sigma)$ or $\mathcal{N}(\mu_2, \Sigma)$ with equal probability,

with $\mu_1 = (-3/4, -3)$, $\mu_2 = (3/4, 3)$ and $\Sigma = I$. If $y = 2$, (x^1, x^2) are drawn from two normal distributions with equal probability, with $\mu_1 = (3, -3)$, $\mu_2 = (-3, 3)$ and the same Σ.

Four settings with different sizes of the training set ($|\mathscr{D}_{trn}|$=200, 90, 70, or 40) are considered to investigate the influence of the sparseness of the data set on the performance of the feature selection methods. In all four settings, m is chosen to be 6 by the cross-validation process.

Table 3.4.1 presents the number of trials (out of 30 trials on different realizations) that feature 1 and 2 are successfully ranked as the first and second most important features. The best performance for each case is highlighted bold.

It is easy to see that the advantage of MLP-FSPP-RFE over other benchmark methods is evident when the feature selection problem becomes more challenging (as the size of training set gets smaller). First, as seen from Table 3.4.1, both filter methods FisherS and MutualI completely fail to identify two key features even in the easiest case (with 200 training samples). This is not surprising because features 1 and 2 alone has nearly no discriminating capability and any filter method that treats features individually will not work on such problem. Therefore, the experiments of these two filter methods on more challenging settings (with less training samples) are omitted. Second, Table 3.4.1 also indicates that MLP-FSPP-RFE outperforms MOI and the difference in performance is especially evident when the learning problem gets harder (with less training samples).

The test error rates varying with the number of top-ranked features as in Figure.3.2 again

Method	$\lvert\mathscr{D}_{trn}\rvert = 200$	$\lvert\mathscr{D}_{trn}\rvert = 90$	$\lvert\mathscr{D}_{trn}\rvert = 70$	$\lvert\mathscr{D}_{trn}\rvert = 40$
MLP-FSPP-RFE	**30**	**30**	**30**	**30**
FisherS	0	$--$	$--$	$--$
MutualI	0	$--$	$--$	$--$
MOI	**30**	29	24	0

Table 3.1: The number of realizations that feature $1, 2$ are successfully ranked in the top two positions over 30 realizations for Weston Problem.

shows that MLP-FSPP-RFE outperforms other methods, especially when the feature selection problem becomes more challenging (as the size of training set gets smaller). The statistical significance of this performance difference is also verified by afore-mentioned paired *t*-tests. When the training set size is small (i.e. 40 or 70) and only the first two top-ranked features are used, the *p*-value obtained is less than 0.05.

It is also worthy to note that MLP-FSPP-RFE consistently produces a test-error curve (Figure.3.2) that has the minimum point when top two features are given. This points to the effectiveness of the proposed feature selection method in removing irrelevant features even when it operates far from the optimum number of feature. This is not the case for the MOI method, as shown in Figure. 3 (c) and (d).

Synthetic Corral Data Sets

In this section, synthetic Corral data set (Corral-6) proposed by Corral [42] and its variants (Corral-46 and Corral-47) proposed by Yu and Liu [89] are used to test the capability of feature selection methods in handling both irrelevant and redundant features. In each of three data sets there are 128 samples. All three data sets (Corral-6, Corral-46 and Corral-47) have four same mutually-independent important boolean features,

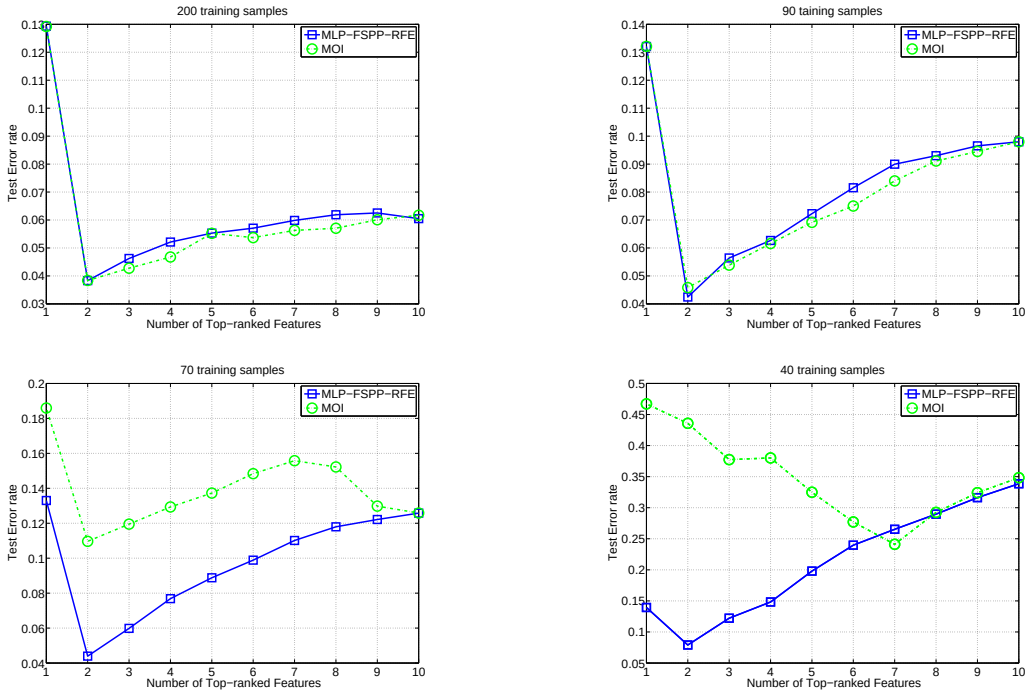


Figure 3.2: Average test error against top-ranked features over 30 realizations of Weston data sets for four training set sizes.

$\{A0, A1, B0, B1\}$, and the same target concept, $y = (A0 \cap A1) \cup (B0 \cap B1)$, but differ in the choices of the other redundant and irrelevant features. The Corral-6 data set contains two other features: an irrelevant feature I taking values from a uniformly random distribution and a redundant feature which matches the target concept 75% of the time and mismatches 25% of the time. Corral-46 contains 28 redundant features and 14 irrelevant features. The 28 redundant features are obtained from the original 4 boolean features (7 redundant features for each of $A0$,$A1$,$B0$ and $B1$) at various correlations levels (1, $15/16$, $14/16$, $\cdots$, $10/16$). These 7 features are correspondingly denoted with a subscript of an increasing number, for example, the 7 redundant features derived from $A0$ include $A0_0, A0_1, \cdots A0_6$. Among the 14 irrelevant features, only two features are uniformly random and each of the remaining 12 is completely correlated with either of these two. Corral-47 is exactly same as Corral-46 except that the former contains one

more redundant feature $R75$. Thus, optimal features sets (after removing all irrelevant and redundant features) for these three data sets should only contain 4 relevant features indeed, as shown in Table 3.4.1.

The feature selection performances of MLP-FSPP-RFE, FisherS, MutualI and MOI on these three synthetic data sets are obtained from 30 realizations with softmax-based MLP. Similar to the experiments in Weston problem, Table 3.4.1 presents the numbers of realizations that optimal features sets are successfully ranked in the top four positions in 30 different realizations. In this table, $\mathscr{I}_G$ refers to the known optimal features sets. For Corral-46 and Corral -47, each optimal feature in $\mathscr{I}_G$ has its duplication in bracket, so only either of them can be selected in optimal feature set. It is easy to see the advantage of the proposed method over benchmark methods in handling both irrelevant and redundant features from this table. Two filter methods, FisherS and MutualI, again almost completely fail to identify optimal features set, while MOI performs well on Corral-6 but poorly on Corral-46 and Corral-47 when more irrelevant and redundant features are adulterated. In contrast to these benchmark methods, MLP-FSPP-RFE consistently performs well in all the three data sets.

The graphs of test error rates against the number of top-ranked features in Figure.3.3 again show better performance of MLP-FSPP-RFE than those of the benchmark methods. This performance advantage can also be verified by the afore-mentioned t-test between MLP-FSPP-RFE and each of the benchmark methods. For example, consider the two relatively more challenging problems of Corral-46 and Corral-47, the p-value obtained is less than 0.05 by comparing the test error rates with optimal feature reduction

	Corral-6	Corral -46	Corral -47
	$A0\ A1$	$A0(A0_0)\ A1(A1_0)$	$A0(A0_0)\ A1(A1_0)$
Method$\setminus\mathscr{I}_G$	$B0\ B1$	$B0(B0_0)\ B1(B1_0)$	$B0(B0_0)\ B1(B1_0)$
MLP-FSPP-RFE	**30**	**30**	**30**
FisherS	0	0	0
MutualI	0	0	0
MOI	**30**	9	10

Table 3.2: The number of realizations that optimal features are successfully ranked in the top four positions over 30 realizations for Corral Problems.

(i.e. when only 4 top-ranked features are left).

3.4.2 Real-world Data Sets

	$\lvert\mathscr{D}_{trn}\rvert$	$\lvert\mathscr{D}_{tst}\rvert$	d	c	m	n_r
Abalone	3133	1044	8	3	11	1
WBCD	350	333	9	2	10	1
Wine	120	58	13	3	13	1
Vehicle	423	423	18	4	4	1
Image	210	2100	19	7	2	1
Waveform	400	4600	21	3	3	1
HillValey	606	606	100	2	2	10
Musk	330	146	166	2	3	10

Table 3.3: Description of real-world data sets for classification problems.

Eight real-world data sets are taken from the UCI machine learning repository [1] and their descriptions are given in Table 3.3, where d, c, m, n_r refer to number of features, number of classes, number of hidden neurons used in the MLP and the number of features removed each time by Algorithm 1, respectively. The Abalone data set has been transformed into a 3-class classification problem following the procedure by David *et. al.* [16]. Figures 3.4-3.11 show the average test error rates against the number of top-ranked features used in the classification for Abalone, WBCD, Wine, Vehicle, Waveform, Image, HillValey and Musk respectively. Results of paired t-test between MLP-

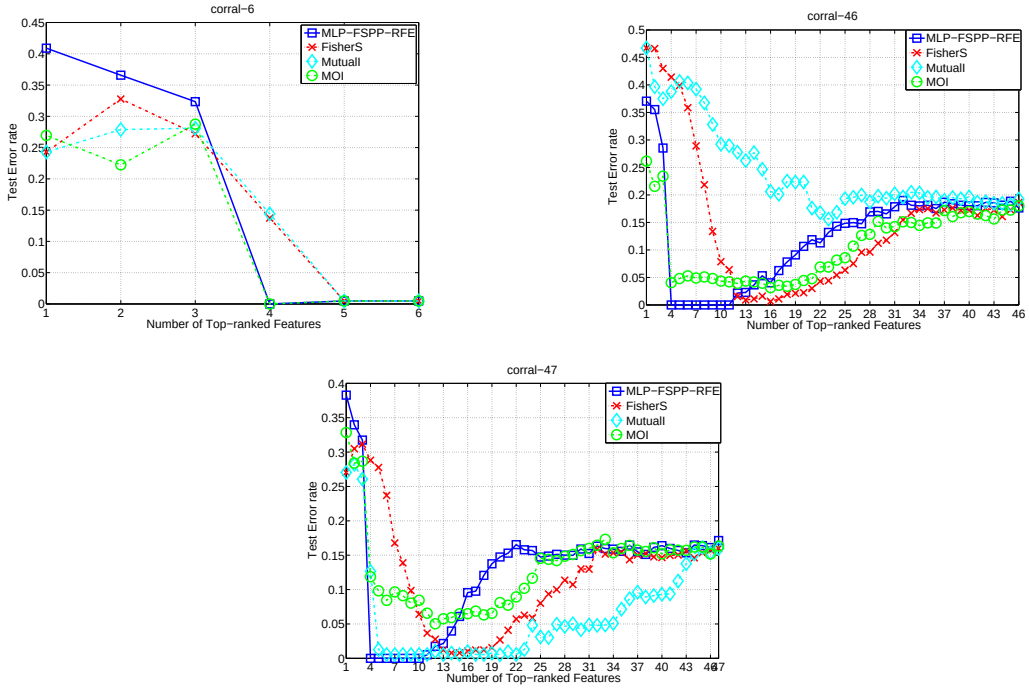


Figure 3.3: Average test error against top-ranked features over 30 realizations of three Corral data sets: (a) Corral-6. (b) Corral-46. (c) Corral-47.

FSPP-RFE and each of benchmark methods are respectively tabulated from Tables 3.4 to 3.11, in which No. is the number of top ranked features and the *p*-values less than 0.05 are highlighted in bold.

For problem Abalone, Figure 3.4 shows the average test error rates against the number of top-ranked features in MLP for both proposed and benchmark methods. It can be observed in this figure that given the same level of the feature selection (with the same number of features removed), MLP-FSPP-RFE generally yields lower average test error rates than benchmark methods. This is confirmed by the paired *t*-tests' result given in Table 3.4. Generally, MLP-FSPP-RFE consistently performs at least as well, if not better than benchmark methods with a few exceptions happen: e.g, in the first row (with only the top-ranked feature left), the test error rate of MLP-FSPP-RFE is significantly

higher than those of FisherS and MutualI. This is not considered as a worrying sign, because they only happen when features are over-eliminated after removing many relevant features in RFE. Usually, early stopping of RFE should have been triggered by the dramatic increase of the test error rate.

For other real-world problems (WBCD, Wine, Vehicle, Image, Waveform, HillValey and Musk), the experimental results show similar patterns to that of the problem Abalone, as shown in Figures 3.5 to 3.11 and Tables 3.5 to 3.11. Generally, our results on paired t-tests show that the proposed method performs at least as well, if not better, than the benchmark methods.

3.4.3 Discussion

Based on extensive numerical experiments, it appears that the proposed method MLP-FSPP-RFE outperforms other existing methods in the literature, especially when the data set is sparse or when the data set has many redundant features. The better performance of MLP-FSPP-RFE over filter methods, FisherS and MutualI, is expected since filter methods have their inherent theoretic pitfalls as mentioned in Chapter 2, but the better performance of MLP-FSPP-RFE over MOI is interesting and deserves attention. Both MLP-FSPP-RFE and MOI use the RFE approach but differ in their ranking criteria. The former uses the "aggregate" sensitivity of MLP probabilistic outputs with respect to a feature over the feature space while the latter relies on a heuristically assigned credit of every feature's contribution to output information.

The better performance of MLP-FSPP-RFE over MOI is related to posterior probability being a better measure of performance over output information. Indeed, the decision function (3.4) of MLP is directly related to the posterior probability but MOI uses the indirect measure of output information. In addition, $\hat{y}$ of $I(y;\hat{y})$ in the MOI method is a discrete variable and, thus, is less discriminating than the continuous nature of the posterior probability. These two factors are likely to be significant when the training data is sparse. The proposed criterion also has a slight edge over MOI in terms of computational cost. As mentioned in Section 3.2, the computational cost of ranking the d features (ignoring training cost) for the proposed method is about $O(N|W|)$ while that of MOI is about $O(2N|W|)$ [72].

3.5 Summary

This chapter proposes a new feature selection method and its numerical evaluation for MLP neural networks. The proposed method is based on the sensitivity of the probabilistic output of the MLP with respect to a given feature. Numerical experiments using the proposed method and other feature selection methods are conducted on several artificial and real-world data sets. In all the experiments, statistical testing shows that the proposed method performs generally better than the other feature selection methods. The proposed method performs particular well for data sets with low samples-to-feature ratios and data sets adulterated with different levels of redundant features. This better performance is very likely due to posterior probability being directly related to the decision function of the MLP and the aggregate of this probabilistic output over the entire

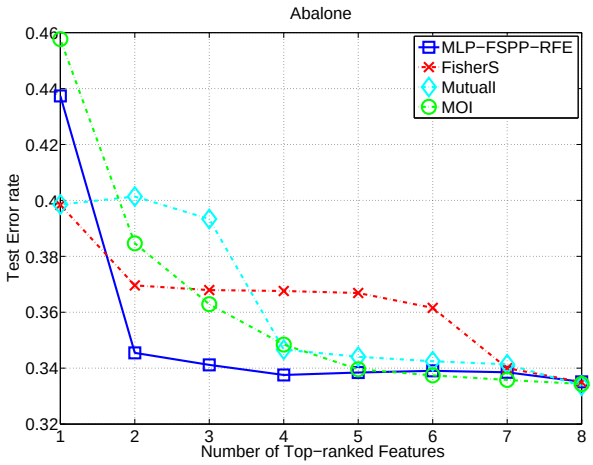


Figure 3.4: Test error rates on Abalone data set

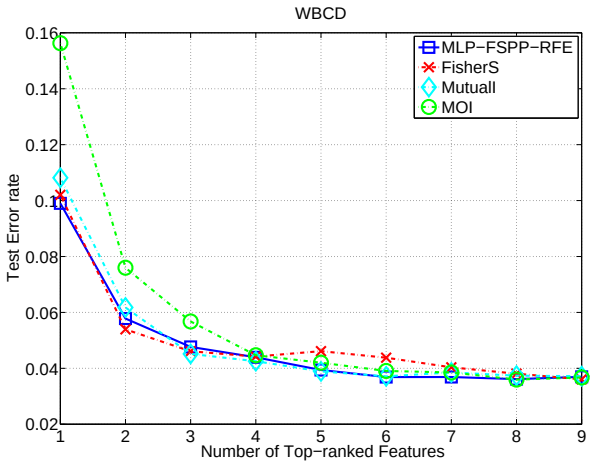


Figure 3.5: Test error rates on WBCD data set

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	43.74	39.84	**0.00-**	39.85	**0.00-**	45.77	**0.00+**
2	34.55	36.96	**0.00+**	40.14	**0.00+**	38.46	**0.00+**
3	34.12	36.79	**0.00+**	39.33	**0.00+**	36.29	**0.00+**
4	33.76	36.76	**0.00+**	34.66	**0.03+**	34.84	**0.02+**
5	33.84	36.69	**0.00+**	34.41	0.13	33.96	0.78
6	33.90	36.16	**0.00+**	34.25	0.30	33.74	0.62
7	33.85	34.00	0.62	34.14	0.37	33.58	0.42
8	33.51	33.50	0.98	33.38	0.69	33.43	0.82

Table 3.4: *t*-test on Abalone data set.

feature space. In addition, the proposed method requires only modest computations.

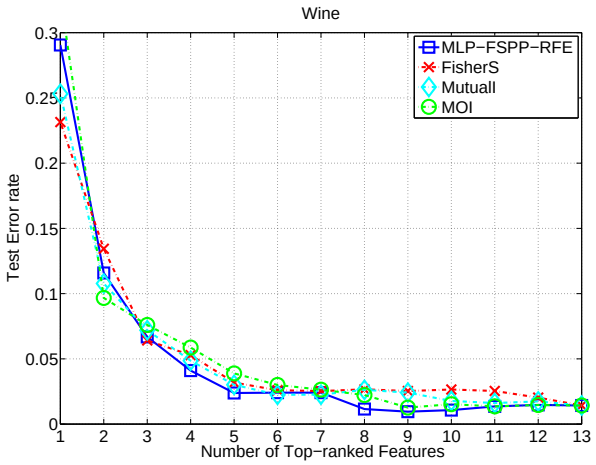


Figure 3.6: Test error rates on Wine data set

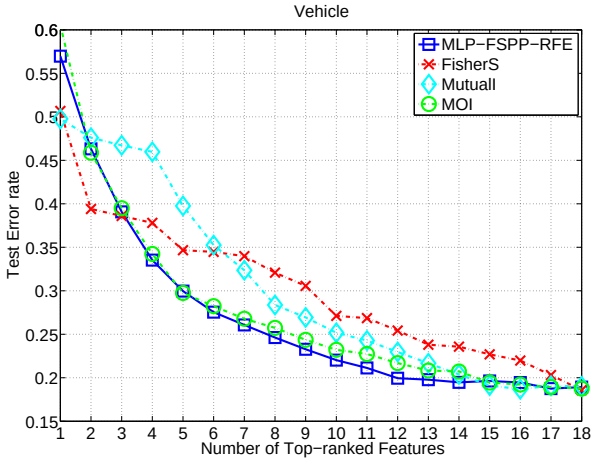


Figure 3.7: Test error rates on Vehicle data set

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	9.90	10.21	0.46	10.82	**0.01+**	15.63	**0.00+**
2	5.78	5.39	0.15	6.18	0.21	7.60	**0.00+**
3	4.77	4.61	0.56	4.51	0.31	5.68	**0.01+**
4	4.40	4.42	0.93	4.26	0.62	4.47	0.80
5	3.94	4.61	**0.01+**	3.89	0.82	4.20	0.29
6	3.69	4.38	**0.00+**	3.71	0.90	3.91	0.24
7	3.69	4.04	0.14	3.85	0.39	3.85	0.51
8	3.62	3.81	0.30	3.74	0.53	3.60	0.92
9	3.70	3.61	0.65	3.72	0.91	3.67	0.90

Table 3.5: *t*-test on WBCD data set.

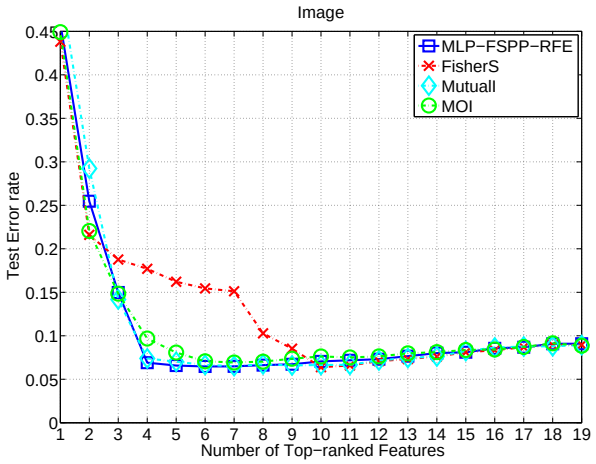


Figure 3.8: Test error rates on Image data set

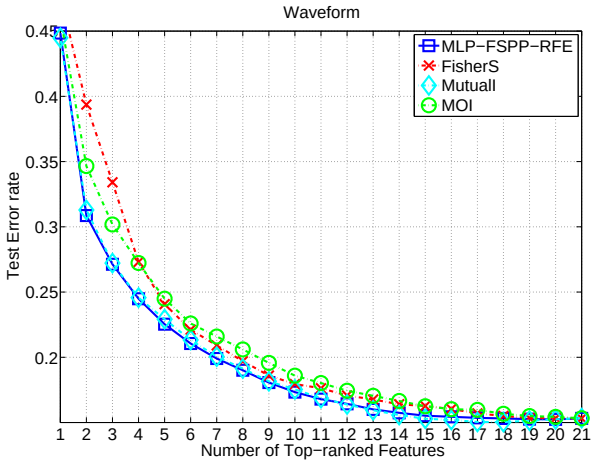


Figure 3.9: Test error rates on Waveform data set

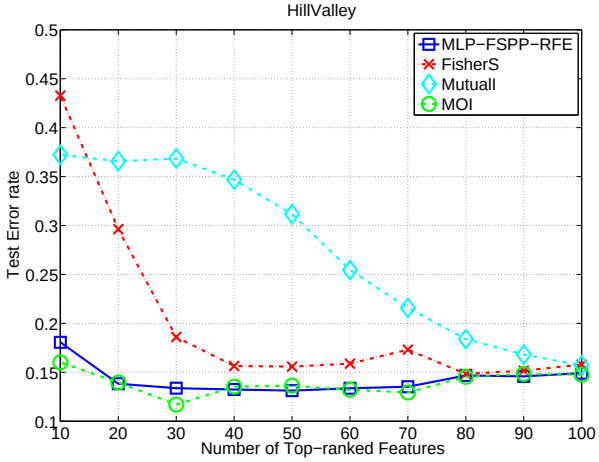


Figure 3.10: Test error rates on HillValley data set

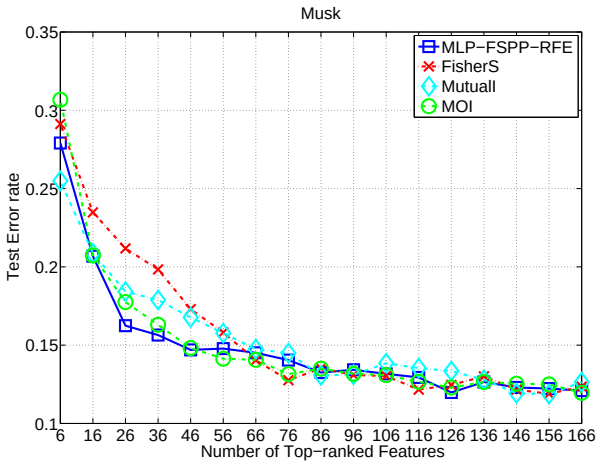


Figure 3.11: Test error rates on Musk data set

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	29.05	23.15	**0.02-**	25.32	0.14	32.92	0.12
2	11.58	13.44	0.09	10.78	0.46	9.67	0.12
3	6.68	6.41	0.67	7.19	0.45	7.59	0.24
4	4.10	5.28	0.07	4.86	0.26	5.86	**0.02+**
5	2.38	3.18	0.13	3.01	0.22	3.88	**0.01+**
6	2.41	2.59	0.71	2.24	0.75	3.00	0.26
7	2.41	2.53	0.79	2.26	0.73	2.64	0.61
8	1.15	2.66	**0.00+**	2.62	**0.00+**	2.22	**0.01+**
9	0.95	2.53	**0.00+**	2.41	**0.00+**	1.26	0.38
10	1.07	2.65	**0.00+**	1.77	0.07	1.52	0.29
11	1.35	2.54	**0.01+**	1.61	0.55	1.36	1.00
12	1.47	2.03	0.20	1.74	0.54	1.46	0.99
13	1.43	1.43	1.00	1.43	1.00	1.43	1.00

Table 3.6: *t*-test on Wine data set.

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	56.97	50.67	**0.00-**	49.77	**0.00-**	60.54	**0.02+**
2	46.32	39.40	**0.00-**	47.60	0.24	45.87	0.78
3	39.07	38.61	0.60	46.72	**0.00+**	39.49	0.72
4	33.53	37.80	**0.00+**	45.97	**0.00+**	34.24	0.44
5	29.97	34.66	**0.00+**	39.76	**0.00+**	29.75	0.77
6	27.54	34.47	**0.00+**	35.25	**0.00+**	28.23	0.37
7	26.08	33.98	**0.00+**	32.36	**0.00+**	26.79	0.24
8	24.62	32.08	**0.00+**	28.37	**0.00+**	25.74	0.14
9	23.28	30.56	**0.00+**	26.94	**0.00+**	24.36	0.06
10	22.02	27.11	**0.00+**	25.17	**0.00+**	23.21	**0.02+**
11	21.13	26.86	**0.00+**	24.27	**0.00+**	22.73	**0.01+**
12	19.95	25.43	**0.00+**	22.96	**0.00+**	21.68	**0.00+**
13	19.78	23.80	**0.00+**	21.66	**0.00+**	20.83	0.06
14	19.47	23.57	**0.00+**	20.38	0.06	20.72	**0.02+**
15	19.63	22.69	**0.00+**	19.11	0.26	19.45	0.71
16	19.45	21.98	**0.00+**	18.69	**0.04+**	19.22	0.57
17	18.75	20.31	**0.00+**	19.08	0.46	18.93	0.66
18	18.93	18.58	0.38	19.00	0.87	18.75	0.67

Table 3.7: *t*-test on Vehicle data set.

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	45.93	43.81	0.06	50.24	**0.00+**	44.91	0.41
2	25.47	21.61	**0.00-**	29.23	**0.01+**	22.05	**0.01-**
3	14.98	18.77	**0.00+**	14.18	0.38	14.84	0.90
4	6.90	17.70	**0.00+**	7.4	0.39	9.68	**0.00+**
5	6.58	16.21	**0.00+**	7.01	0.14	8.07	**0.02+**
6	6.47	15.45	**0.00+**	6.65	0.54	7.07	0.09
7	6.49	15.11	**0.00+**	6.51	0.93	6.92	0.15
8	6.63	10.28	**0.00+**	6.71	0.74	7.03	0.20
9	6.72	8.52	**0.01+**	6.59	0.62	7.36	**0.04+**
10	7.06	6.39	0.05	6.65	0.14	7.63	0.08
11	7.18	6.58	0.07	6.67	0.13	7.51	0.36
12	7.31	7.09	0.43	7.16	0.62	7.63	0.28
13	7.66	7.32	0.33	7.45	0.53	7.98	0.38
14	8.01	7.61	0.22	7.62	0.17	8.12	0.73
15	8.11	8.10	0.98	8.24	0.71	8.37	0.45
16	8.55	8.33	0.44	8.65	0.76	8.42	0.68
17	8.68	8.68	0.99	8.76	0.82	8.71	0.94
18	9.08	8.93	0.71	8.80	0.45	9.13	0.89
19	9.09	8.96	0.75	8.92	0.69	8.86	0.59

Table 3.8: *t*-test on Image data set.

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
1	44.85	47.71	**0.00**+	44.48	0.41	46.10	**0.02**+
2	30.86	39.37	**0.00**+	31.29	0.28	34.65	**0.00**+
3	27.13	33.41	**0.00**+	27.22	0.82	30.17	**0.00**+
4	24.48	27.31	**0.00**+	24.57	0.72	27.22	**0.00**+
5	22.53	24.07	**0.00**+	22.92	0.12	24.49	**0.00**+
6	21.05	22.11	**0.00**+	21.33	0.24	22.59	**0.00**+
7	19.90	20.91	**0.00**+	20.05	0.46	21.59	**0.00**+
8	19.01	19.70	**0.00**+	19.14	0.45	20.60	**0.00**+
9	18.06	18.60	**0.00**+	18.16	0.54	19.57	**0.00**+
10	17.33	17.94	**0.00**+	17.54	0.13	18.59	**0.00**+
11	16.79	17.62	**0.00**+	16.87	0.57	18.03	**0.00**+
12	16.43	17.07	**0.00**+	16.38	0.75	17.43	**0.00**+
13	16.01	16.76	**0.00**+	15.94	0.63	17.04	**0.00**+
14	15.74	16.40	**0.00**+	15.57	0.31	16.65	**0.00**+
15	15.52	16.26	**0.00**+	15.29	0.13	16.24	**0.00**+
16	15.44	16.03	**0.00**+	15.20	0.14	16.02	**0.00**+
17	15.35	15.70	0.06	15.04	0.06	15.97	**0.00**+
18	15.31	15.50	0.28	15.03	0.08	15.67	**0.03**+
19	15.26	15.34	0.64	15.14	0.43	15.49	0.12
20	15.24	15.33	0.57	15.24	0.99	15.43	0.18
21	15.32	15.32	1.00	15.33	0.90	15.32	0.95

Table 3.9: *t*-test on Waveform data set.

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
10	18.07	43.28	**0.00**+	37.23	**0.00**+	16.01	0.30
20	13.83	29.63	**0.00**+	36.57	**0.00**+	13.96	0.91
30	13.38	18.60	**0.01**+	36.85	**0.00**+	11.70	0.13
40	13.24	15.64	0.26	34.69	**0.00**+	13.53	0.78
50	13.13	15.58	0.21	31.16	**0.00**+	13.63	0.64
60	13.36	15.88	0.22	25.45	**0.00**+	13.21	0.90
70	13.53	17.31	0.07	21.58	**0.00**+	12.93	0.65
80	14.67	14.84	0.90	18.39	**0.01**+	14.54	0.90
90	14.59	15.18	0.65	16.81	0.10	14.84	0.84
100	14.91	15.78	0.46	15.70	0.50	14.76	0.90

Table 3.10: *t*-test on Hillvalley data set.

No.	MLP-FSPP-RFE	FisherS		MutualI		MOI	
	mean ERR	mean ERR	p-value	mean ERR	p-value	mean ERR	p-value
6	27.91	29.13	0.34	25.49	**0.05**-	30.67	0.05
16	20.68	23.48	**0.00**+	20.87	0.85	20.73	0.96
26	16.24	21.18	**0.00**+	18.42	**0.02**+	17.75	0.12
36	15.64	19.84	**0.00**+	17.89	**0.01**+	16.30	0.36
46	14.70	17.30	**0.00**+	16.77	**0.02**+	14.82	0.89
56	14.78	15.81	0.22	15.77	0.22	14.13	0.44
66	14.51	14.03	0.65	14.75	0.79	14.07	0.56
76	14.05	12.76	0.20	14.49	0.61	13.15	0.32
86	13.25	13.52	0.76	13.07	0.83	13.50	0.79
96	13.43	13.09	0.66	13.12	0.69	13.17	0.73
106	13.17	13.03	0.87	13.85	0.44	13.09	0.93
116	12.94	12.16	0.33	13.54	0.51	12.66	0.75
126	11.98	12.46	0.57	13.35	0.12	12.29	0.72
136	12.65	13.01	0.66	12.79	0.87	12.64	1.00
146	12.30	12.16	0.86	11.93	0.66	12.53	0.77
156	12.23	11.89	0.66	11.88	0.66	12.49	0.73
166	12.10	12.35	0.72	12.64	0.46	11.96	0.83

Table 3.11: *t*-test on Musk data set.

Chapter 4

Feature Selection via Sensitivity Analysis of SVR Probabilistic Outputs

This chapter proposes a new wrapper-based feature selection method for support vector regression (SVR). Under the probabilistic framework, the output of a standard SVR can be interpreted as $p(y|x)$, the conditional density function of target $y \in \mathbb{R}$ given input $x \in \mathbb{R}^d$ for a given data set. The proposed method relies on the sensitivity of $p(y|x)$ with respect to a given feature as a measure of importance of this feature. More exactly, the importance score of a feature is the aggregation, over the feature space, of the difference of $p(y|x)$ with and without the feature. The exact computations of the proposed method is expensive, two approximations are proposed. Each of the two approximations, embedded in an overall feature selection scheme, is tested on various artificial and real-world data sets and compared with several other existing feature selection methods. The experimental result shows that the proposed method performs generally better than,

if not at least as well as, other methods in almost all experiments.

This chapter is organized as follow: Section 4.1 reviews the formulas of SVR with probabilistic outputs. Section 4.2 presents details of the proposed feature ranking criterion and the two approximations. Section 4.3 shows the overall feature selection scheme. Results of numerical experiment of the proposed method, benchmark against other methods, are reported in Section 4.4. Section 4.5 summarizes the chapter.

4.1 Preliminary

The expressions of standard SVR are reviewed in 2.1.2 of Chapter 2. However, the output function of SVR, as shown in (2.16), provides an estimate, $f(x)$, for output y for any x but provides no information on the confidence level of this estimate. Recognizing this shortcoming, several attempts to incorporate probabilistic values to SVR output have been reported in the literature. Following the approach of Bayesian framework for neural network [54], Law and Kwok [48] propose a Bayesian support vector regression (BSVR) formulation incorporating probabilistic information. Gao et al. [26] improve upon BSVR by deriving the evidence and error bar approximation. Chu et al.[14] propose the use of a unified loss function over the standard ε-insensitive loss function and provide better accuracy in evidence evaluation and inferences.

Another approach to obtaining probabilistic output of the regressor is that used in the Neural Networks framework [5]. It assumes that the output of the regressor is corrupted

with noise in the form of

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

where δ belongs to the Gaussian distribution. Lin and Weng [51] also consider the case where δ belongs to the Laplace distribution. Equivalently, this means that density functions of y for a given x are

$$p^L(y|x;\sigma) = \frac{1}{2\sigma}\exp(-\frac{|y-f(x)|}{\sigma}), \tag{4.2}$$

$$p^G(y|x;\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(y-f(x))^2}{2\sigma^2}) \tag{4.3}$$

for the Laplace and Gaussian cases respectively. Like the Neural Network approach, the intention is to obtain estimates of σ of (4.2) and (4.3) from $\mathscr{D}$. If $p(x,y)$ is the joint density function of x and y, the likelihood function, as a function of σ, of observing $\mathscr{D}$ is given by

$$L(\sigma) = \Pi_{i\in\mathscr{I}_\mathscr{D}} p(x_i,y_i) = \Pi_{i\in\mathscr{I}_\mathscr{D}} p(y_i|x_i;\sigma)p(x_i),$$

under the assumption of independent and identically distributed samples. By further assuming that $p(x)$ is independent of σ, the expressions of σ can be obtained by maximizing the logarithm function of $L(\sigma)$ [5, 20]. These expressions are

$$\sigma^L = \frac{\sum_{i\in\mathscr{I}_\mathscr{D}} |y_i - f(x_i)|}{N}, \tag{4.4}$$

$$(\sigma^G)^2 = \frac{\sum_{i\in\mathscr{I}_\mathscr{D}} (y_i - f(x_i))^2}{N} \tag{4.5}$$

for the Laplace and Gaussian distributions respectively. It has been shown [51] that this approach is competitive in terms of performance to the BSVR methods.

4.2 The Proposed Wrapper-based Feature Selection Criterion for Regression

For regression problems, the proposed feature selection method evaluating feature importance relies on measures of difference between two density functions. Our choice of this measure is the well-known Kullback-Leibler divergence (KL divergence), $D_{KL}(\cdot;\cdot)$. Given two distributions $p(y)$ and $q(y)$,

$$D_{KL}(p(y);q(y)) = \int p(y)\log\frac{p(y)}{q(y)}dy. \tag{4.6}$$

From its definition, it is easy to verify that $D_{KL}(p(y);q(y)) \geq 0$ for any $p(y)$ and $q(y)$, $D_{KL}(p(y);q(y)) = 0$ if and only if $p(y) = q(y)$ and $D_{KL}(p(y);q(y))$ is not symmetrical with respect to its arguments. The last property is a result of treating $p(y)$ as the reference distribution. In cases where symmetry of the arguments is important or that a reference distribution does not exist, modifications to $D_{KL}(\cdot;\cdot)$ can be easily achieved.

In the case of SVR, the density function $p(y|x)$ at any x is assumed to be (4.2) or (4.3) with $f(\cdot)$ being the solution obtained from (2.16). Given $x \in \mathbb{R}^d$, $x_{-j} \in \mathbb{R}^{d-1}$ can be obtained by removing the j^{th} feature from x, or, equivalently, $x_{-j} = Z_j^d x$. With this, the difference of the two density functions $p(y|x)$ and $p(y|x_{-j})$ at a particular x (and

hence x_{-j}) is $D_{KL}(p(y|x);p(y|x_{-j}))$. The proposed feature importance measure is an aggregation of $D_{KL}(p(y|x);p(y|x_{-j}))$ over all x in the x space. More exactly, the measure is

$$S_D(j) = \int D_{KL}(p(y|x);p(y|x_{-j}))p(x)dx. \tag{4.7}$$

The motivation for defining S_D is simple: the greater the D_{KL} divergence between $p(y|x)$ and $p(y|x_{-j})$ over the x space, the greater the importance of the j^{th} feature. For convenience, (4.7) is termed SD measure, short for Sensitivity of Density Functions.

In (4.7), $p(y|x)$ is either (4.2) or (4.3) with the prediction function $f(\cdot)$ trained on $\mathscr{D}$. Similarly, $p(y|x_{-j})$ is obtained from the SVR output function trained using the derived dataset $\mathscr{D}_{-j} := \{(x_{-j,i},y_i)|x_{-j,i} = Z_j^d x_i \text{ for all } (x_i,y_i) \in \mathscr{D}\}$. Thus, evaluations of $S_D(j)$, $j = 1,\cdots,d$ require the training of SVR d times, each with a different $\mathscr{D}_{-j}$. Clearly, this is a computationally expensive process. Like the procedure mentioned in Section 3.2 of Chapter 3, a random permutation (RP) process [8, 59] is used to approximate $p(y|x_{-j})$ such that the retraining of SVR is avoided.

Let $x_{(j)} \in \mathbb{R}^d$ be the sample derived from x after the RP process on the j^{th} feature and let $p(y|x_{(j)})$ be the conditional density function of y given $x_{(j)}$. Then, we can get a theorem analogous to theorem 3.2.1 below.

Theorem 4.2.1.

$$p(y|x_{(j)}) = p(y|x_{-j}) \tag{4.8}$$

Therefore, the density function $p(y|x_{-j})$ of (4.7) can be replaced by $p(y|x_{(j)})$. Such a replacement brings about significant computational advantage since $p(y|x_{(j)})$ can be evaluated from (4.2) or (4.3) using $f(x_{(j)})$ obtained from the SVR training using $\mathscr{D}$. By assuming that $p(y|x_{(j)})$ can be evaluated from (4.2) or (4.3) using $f(x_{(j)})$ obtained from the SVR training using $\mathscr{D}$ (since x and $x_{(j)}$ are both d-dimensional), this avoids the expensive d-time retraining of SVR on $\mathscr{D}_{-j}$. Correspondingly, (4.7) can be equivalently stated as:

$$S_D(j) = \int D_{KL}(p(y|x); p(y|x_{(j)}))p(x)dx. \tag{4.9}$$

Figure 4.1 shows a plot of $p(y_i|x_i)$ and $p(y_i|x_{(j),i})$ at one choice of x_i for a typical SVR problem with $d = 1$. To compute the S_D, further approximation of (4.9) is needed, resulting in

$$\hat{S}_D(j) = \frac{1}{N} \sum_{i \in \mathscr{I}_{\mathscr{D}}} D_{KL}(p(y_i|x_i); p(y_i|x_{(j),i})). \tag{4.10}$$

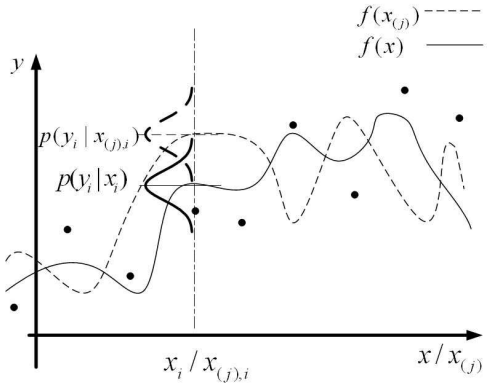


Figure 4.1: Demonstration of the proposed feature ranking criterion with $d = 1$. Dots indicate locations of y_i

When $p(y|x)$ and $p(y|x_{(j)})$ are Laplace functions or Gaussian functions, explicit expressions of $\hat{S}_D(j)$ exist. Using (4.2) and following the derivation in Appendix B, the KL divergence for the case of Laplace function can be shown to be,

$$D_{KL}(p^L(y|x;\sigma^L);p^L(y|x_{(j)};\sigma^L_{(j)})) = \ln\frac{\sigma^L_{(j)}}{\sigma^L} - 1 + \frac{\sigma^L}{\sigma^L_{(j)}}\exp(-\frac{|f(x)-f(x_{(j)})|}{\sigma^L}) + \frac{|f(x)-f(x_{(j)})|}{\sigma^L_{(j)}} \tag{4.11}$$

for a given x where σ^L is that given by (4.4) and $\sigma^L_{(j)}$ is obtained from (4.4) by replacing $f(x)$ with $f(x_{(j)})$. Using (4.11) in (4.10) and removing associated constants yields

$$\hat{S}^L_D(j) = \frac{1}{N}\sum_{i\in\mathscr{I}_\mathscr{D}}\left[\frac{\sigma^L}{\sigma^L_{(j)}}\exp(-\frac{|f(x_i)-f(x_{(j),i})|}{\sigma^L}) + \frac{|f(x_i)-f(x_{(j),i})|}{\sigma^L_{(j)}} + \ln\frac{\sigma^L_{(j)}}{\sigma^L}\right]. \tag{4.12}$$

Following the same development for the case when $p(y|x)$ is Gaussian, the expressions are

$$D_{KL}(p^G(y|x;\sigma^G);p^G(y|x_{(j)};\sigma^G_{(j)})) = \ln\frac{\sigma^G_{(j)}}{\sigma^G} + \frac{f(x)^2 + f(x_{(j)})^2 + (\sigma^G)^2 - 2f(x)f(x_{(j)})}{2(\sigma^G_{(j)})^2} - \frac{1}{2} \tag{4.13}$$

and

$$\hat{S}^G_D(j) = \frac{1}{2N}\sum_{i\in\mathscr{I}_\mathscr{D}}\left[\frac{(f(x_i)-f(x_{(j),i}))^2}{(\sigma^G_{(j)})^2} + (\frac{\sigma^G}{\sigma^G_{(j)}})^2 + 2\ln\frac{\sigma^G_{(j)}}{\sigma^G}\right] \tag{4.14}$$

where the expression of (4.13) is given by [61].

In summary, $\hat{S}_D(j)$ can be computed for all $j = 1, \cdots, d$, after a one-time training of SVR, one-time evaluation of σ^L (or σ^G), d-time RP process, d-time evaluation of $\sigma^L_{(j)}$ (or $\sigma^G_{(j)}$) and d-time evaluation of D_{KL}.

Remark 4.2.1. *The kernel matrix is different for each of the d-time evaluation of $\sigma^L_{(j)}$ (or $\sigma^G_{(j)}$) and this incurs additional computations. Such computations can be kept low using update formulae. Suppose x_r, x_q and $x_{(j),r}, x_{(j),q}$ are two samples before and after the RP process is applied to feature j. It is easy to show that $K(x_{(j),r}, x_{(j),q}) = K(x_r, x_q) + x^j_{(j),r} * x^j_{(j),q} - x^j_r * x^j_q$ for linear kernel and $K(x_{(j),r}, x_{(j),q}) = K(x_r, x_q) * \exp[\kappa(x^j_r - x^j_q)^2 - \kappa(x^j_{(j),r} - x^j_{(j),q})^2]$ with kernel parameter κ for Gaussian kernel.*

4.3 Feature Selection Scheme

Analogous to the analysis in Section 3.3, the proposed $\hat{S}^L_D$ and $\hat{S}^G_D$ can be used in two ways: 1.) it yields a ranking list of all features based on a one time training of SVR on $\mathscr{D}$; 2.) it yields a ranking list of all features based on the recursive feature elimination (RFE) scheme. In each iteration of RFE, a ranking of all remaining features is obtained using some appropriate measures ($\hat{S}^L_D$, $\hat{S}^G_D$ or others). The least important feature, as determined by the measure is then removed from further consideration. This procedure stops after $d - r$ iterations to yield the top r features. Accordingly, the overall scheme with respect to measure $\hat{S}^L_D$ ($\hat{S}^G_D$) is referred to as SD-L-RFE (SD-G-RFE). Inputs to scheme SD-L-RFE are $\mathscr{D}$ and $\mathscr{I} = \{1, \cdots, d\}$, while the output is a ranked list of features in the form of an index set $\mathscr{I}^f = \{i^f_1, \cdots, i^f_d\}$ where $i^f_j \in \mathscr{I}$ for each $j = 1, \cdots, d$

in decreasing order of importance.

Following Theorem 4.2.1, the associated computational costs of the SD-L-RFE (SD-G-RFE) scheme is the training of SVR at each iteration and the evaluations of $\hat{S}_D^G(j)(\hat{S}_D^L(j))$ using (4.13) ((4.11)) for each j of the remaining features in that iteration. This is the case of the proposed scheme. In the next section where other benchmark methods are discussed, the retraining of SVR at each iteration and within the iteration may be needed for the ranking of features because of inapplicability of Theorem 4.2.1

4.4 Numerical Experiment

This section presents result of numerical experiment of SD-L-RFE, SD-G-RFE and the several existing benchmark methods mentioned in Section 2.2, Mutual Information (MI) based method [53] of (2.31), Dependence Maximization method (HSIC) [74], SVM-RFE ($\Delta\|\omega\|^2$) [33] of (2.37), radius-margin bound based method (RMB) [65] of (2.39) and span bound based method (SpanB) [65] of (2.40), on artificial and real-world data sets. The first two benchmark methods are filter methods while the last three are wrapper methods. All methods, except mutual information method, use the same RFE scheme described in Section 4.3 for ranking the features, and hence they are referred to as mRMR, HSIC-RFE, $\Delta\|\omega\|^2$-RFE, RMB-RFE and SpanB-RFE, respectively.

Note that the retraining of SVR within each RFE iteration is not needed for $\Delta\|\omega\|^2$-RFE. However, in the implementation of RMB-RFE and SpanB-RFE by [65], retraining is used within each iteration of the RFE scheme. Obviously, this is much more expensive

process than the proposed method since the result of Theorem 4.2.1 is not applicable to them. Our experiments include both cases: RMB-RFE and SpanB-RFE when retraining is not used and RMB-RFE* and SpanB-RFE* when it is.

For each data set, the result of the experiment is reported over 30 realizations, following the procedure of Rätsch [67]. As usual, $\mathscr{D}_{trn}$ is used for SVR training, hyper-parameters tuning and feature ranking while $\mathscr{D}_{tst}$ is used for unbiased evaluation of the feature selection performance. For each realization, $\mathscr{D}_{trn}$ is normalized to zero mean and unit standard deviation and its normalization parameters are then used to normalize $\mathscr{D}_{tst}$. The kernel function used for all problems is $K(x_i,x_j) = \exp(-\kappa\|x_i - x_j\|^2)$ where κ is the kernel parameter. In each experiment, all hyper-parameters (C,κ,ε) are chosen by a 5-fold cross-validation on the first five realizations of $\mathscr{D}_{trn}$, and the hyper-parameters corresponding to the lowest average cross-validation error among five realizations is chosen. The grid over the (C,κ,ε) is $[2^{-2},2^{-1},\cdots,2^{6}]\times[2^{-6},2^{-5},\cdots,2^{2}]\times[2^{-5},2^{-4},\cdots,2^{2}]$.

Two well-known regression performance measures, namely mean squared error (MSE) and squared correlation coefficient (SCC), are used to evaluate the performance. They are given by

$$\text{MSE} := \frac{\sum_{i=1}^{|\mathscr{D}_{tst}|}(\hat{y}_i - y_i)^2}{|\mathscr{D}_{tst}|}, \tag{4.15}$$

$$\text{SCC} := \frac{(|\mathscr{D}_{tst}|\sum_{i=1}^{|\mathscr{D}_{tst}|}\hat{y}_i y_i - \sum_{i=1}^{|\mathscr{D}_{tst}|}\hat{y}_i\sum_{i=1}^{|\mathscr{D}_{tst}|}y_i)^2}{(|\mathscr{D}_{tst}|\sum_{i=1}^{|\mathscr{D}_{tst}|}\hat{y}_i^2 - \sum_i \hat{y}_i\sum_{i=1}^{|\mathscr{D}_{tst}|}\hat{y}_i)(|\mathscr{D}_{tst}|\sum_{i=1}^{|\mathscr{D}_{tst}|}y_i^2 - \sum_{i=1}^{|\mathscr{D}_{tst}|}y_i\sum_i y_i)} \tag{4.16}$$

where y_i and $\hat{y}_i$, for $i \in \{1,\cdots,|\mathscr{D}_{tst}|\}$, are the true and predicted target values respectively .

Statistical paired t-test using MSE and SCC are conducted for all problems. Specifically, paired t-test between SD-L-RFE and each of the other methods is conducted using different number of top ranked features. Herein, the null hypothesis is that the mean MSE or SCC of the two tested methods are the same against the alternate hypothesis that they are not. The chance that this null hypothesis is true is measured by the returned p-value and the significance level is set at 0.05 for all experiments. The symbols "$+$" and "$-$" are used to indicate the win or loss situation of SD-L-RFE over the other tested method.

In all experiments, the numerical algorithm for training of SVR is implemented by the LIBSVM package [10], where sequential minimal optimization method is used to solve the dual problem (2.14).

4.4.1 Artificial Problems

In this subsection, three artificial regression problems are used to evaluate the performance of every feature selection method. The first two problems were used in [25], and the last one is new for the purpose of investigating different kinds of interaction among features. Each problem has 10 variables $x^1, \cdots, x^{10}$ and the target variable y depends on some of the features as given in their underlying functions:

- Additive function problem

$$y = 0.1\exp(4x^1) + \frac{4}{1+\exp(-20(x^2-0.5))} + 3x^3 + 2x^4 + x^5 + \delta,$$

- Interactive function problem

$$y = 10\sin(\pi x^1 x^2) + 20(x^3 - 0.5) + 10x^4 + 5x^5 + \delta,$$

- Exponential function problem

$$y = 10\exp(-((x^1)^2 + (x^2)^2)) + \delta,$$

where $x^j,\ \forall j = 1, \cdots, 10$ is uniformly distributed within the range [0,1] for the first two problems and [-1,1] for the last. Gaussian noise $\delta \sim \mathscr{N}(0, 0.1)$ for the first two problems while $\delta \sim \mathscr{N}(0, 0.2)$ for the last.

Each artificial problem has 2000 samples. They are randomly split into $\mathscr{D}_{trn}$ and $\mathscr{D}_{tst}$ in the ratio of $|\mathscr{D}_{trn}|$:$|\mathscr{D}_{tst}|$=1:9. To investigate the effect of sparseness of the training set, decreasing sizes of $|\mathscr{D}_{trn}|$ are also used while $|\mathscr{D}_{tst}|$ is maintained at 1800.

Table 4.4.1 presents the number of realizations (out of 30 realizations) that relevant feature are successfully ranked as the top features by the various methods for the different settings of $|\mathscr{D}_{trn}|$. The best performance in each setting is highlighted in bold. From this table, the advantage of the proposed methods is clear. They generally performs at least as well as if not better than all other benchmark methods except when $|\mathscr{D}_{trn}| = 50$ in the interactive problem. For benchmark methods RMB-RFE* and SpanB-RFE*, the proposed methods yield comparable performance. It is also evident that as the size of $|\mathscr{D}_{trn}|$ decreases, the performance of proposed methods generally degrades less than that of benchmark methods. In fact, SD-L-RFE correctly ranks the important features in the

top two positions for all settings for the exponential function problem.

Figure 4.2 shows the average MSE and SCC against top-ranked features over 30 realizations on $\mathscr{D}_{tst}$ for exponential problem. Methods RMB-RFE and SpanB-RFE are not shown since they completely fail as shown in Table 4.4.1. From this figure, the advantages of the proposed methods are obvious. Specifically, the proposed methods perform better than RMB-RFE* and SpanB-RFE* when $|\mathscr{D}_{trn}| = 100, 70$, better than HSIC-RFE and $\Delta\|\omega\|^2$-RFE when $|\mathscr{D}_{trn}| = 50, 40$, and better than mRMR for all $|\mathscr{D}_{trn}|$. This can be verified by aforementioned t-test. Also, it is interesting to see that the curves yielded by SD-L-RFE and SD-G-RFE have the minimal point when the top two features are selected. These bimodal curves strongly validate the effectiveness of the proposed feature selection methods. This is not the case for other methods. The figures for other two problems show the similar patterns and therefore not shown here.

4.4.2 Real Problems

Six real-world data sets from the Statlib[1], UCI repository [1] and Delve archive[2] are used for evaluation purposes. Description of these data sets and the parameters used in the experiments are given in Table 4.2.

Tables 4.3 to 4.8 show the t-test results for six real-world data sets respectively. It is seen from these tables that the proposed methods consistently perform at least as well, if not better than all benchmark methods and the advantage is more significant for mpg,

[1] http://lib.stat.cmu.edu/datasets/
[2] http://www.cs.toronto.edu/∼delve/data/datasets.html

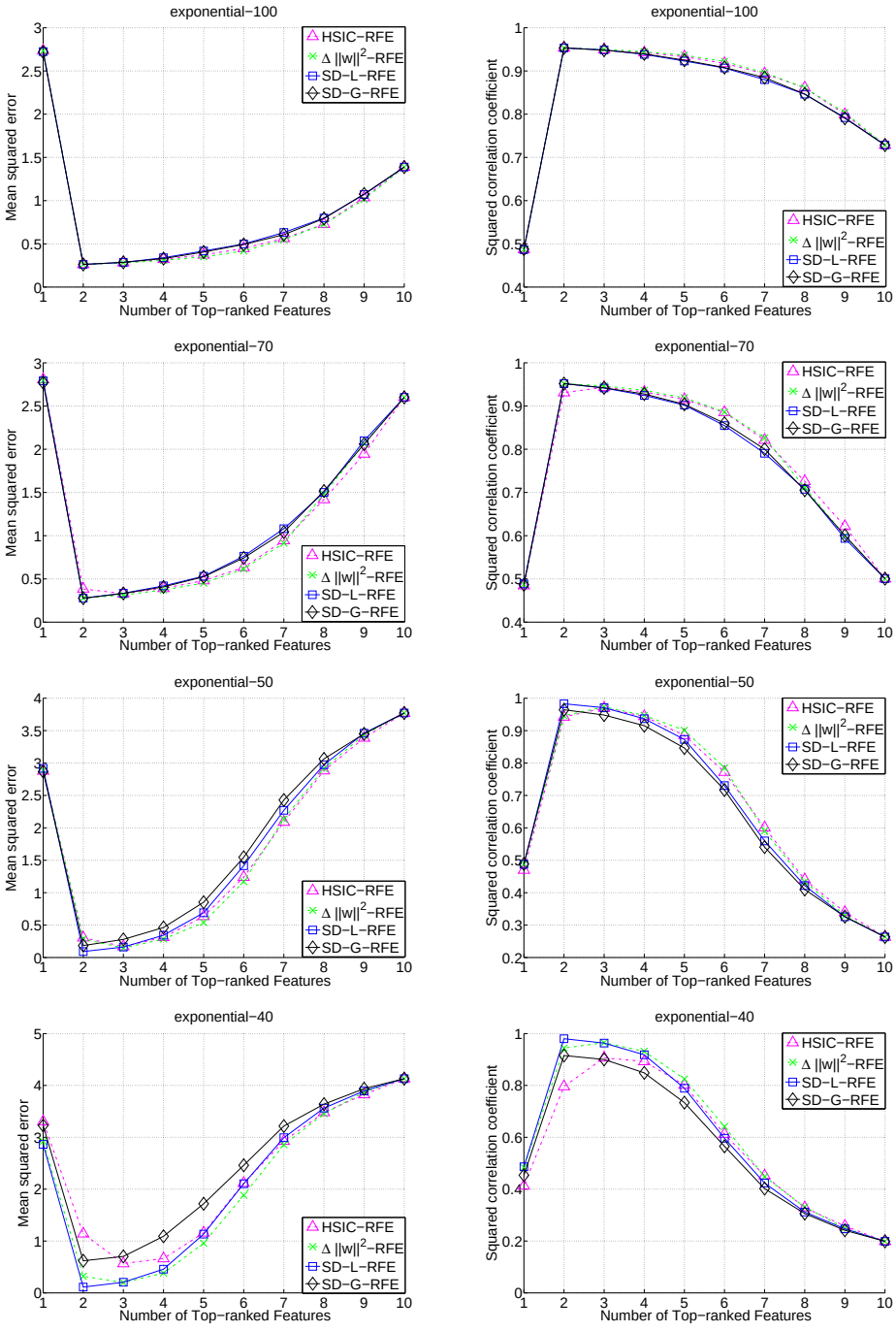


Figure 4.2: Average MSE (left-hand side) and average SCC (right-hand side) against top-ranked features over 30 realizations for Exponential Function Problem with six different settings

	Method\$|\mathscr{D}_{trn}|$	200	100	70	50
	SD-L-RFE	**30**	27	21	**19**
	SD-G-RFE	**30**	**28**	**23**	**19**
	mRMR	19	7	1	0
	HSIC-RFE	14	5	5	3
Additive	$\Delta\|\omega\|^2$-RFE	4	5	11	4
	RMB-RFE	0	0	0	0
	SpanB-RFE	0	1	0	0
	RMB-RFE*	**30**	25	22	9
	SpanB-RFE*	**30**	23	20	9
	Method\$|\mathscr{D}_{trn}|$	200	100	70	50
	SD-L-RFE	**30**	**30**	29	12
	SD-G-RFE	**30**	**30**	**30**	11
	mRMR	9	2	0	0
	HSIC-RFE	7	9	8	6
Interactive	$\Delta\|\omega\|^2$-RFE	0	14	9	10
	RMB-RFE	0	0	0	0
	SpanB-RFE	0	0	0	0
	RMB-RFE*	**30**	**30**	**30**	**20**
	SpanB-RFE*	**30**	**30**	**30**	16
	Method\$|\mathscr{D}_{trn}|$	100	70	50	40
	SD-L-RFE	**30**	**30**	**30**	**30**
	SD-G-RFE	**30**	**30**	29	28
	mRMR	18	2	0	0
	HSIC-RFE	**30**	29	28	22
Exponential	$\Delta\|\omega\|^2$-RFE	**30**	**30**	28	28
	RMB-RFE	0	0	0	0
	SpanB-RFE	0	1	0	1
	RMB-RFE*	4	5	29	27
	SpanB-RFE*	28	28	**30**	29

Table 4.1: The number of realizations that relevant feature are successfully ranked in the top positions over 30 realizations for three artificial problems. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold.

abalone, cpusmall, housing and bodyfat data sets. There are two exceptions: the first few rows of data sets, abalone and bodyfat, show that the SD-L-RFE is statistically worse off than some benchmark methods. This should not be seen as a worrying sign as it happens for the case where one or two features are used. Clearly, this case corresponds to one of over-elimination of features. In practice, early stopping of RFE would have

Data sets	$\lvert\mathscr{D}_{trn}\rvert$	$\lvert\mathscr{D}_{tst}\rvert$	d	C	κ	ε
mpg	353	39	7	2^6	2^{-4}	2
abalone	1254	2923	8	2^6	2^{-5}	2
cpusmall	820	7372	12	2^6	2^{-5}	2
housing	456	50	13	2^6	2^{-4}	2
bodyfat	227	25	14	2^{-2}	2^{-6}	2^{-5}
triazines	168	18	60	2^{-1}	2^{-6}	2^{-3}

Table 4.2: Description of real-world data sets for regression problem.

been triggered by the substantial increase of MSE or decrease of SCC.

4.4.3 Discussion

In summary, the effectiveness of the proposed feature selection method is demonstrated for both artificial and real-world problems. In artificial problems, the proposed method can consistently yield better performance than all three benchmark methods, and the advantage is more evident when $\lvert\mathscr{D}_{trn}\rvert$ is small. This is confirmed by statistical paired t-test results. Furthermore, when the training data become sparse, the performances of the proposed methods degrade much less than the benchmark methods. In real-world problems, it can be observed from all plots and t-test results that the proposed methods consistently perform at least as well, if not better than benchmark methods for all problems.

The better performance of the proposed method over mRMR is expected since this common filter method is not effective in capturing effects of 3 or more interacting features. The other filter method, HSIC-RFE, appears to be quite effective in dealing with data having interacting features, and generally shows nearly comparable performance with the wrapper method $\Delta\|\omega\|^2$-RFE. However, it is not as effective as the proposed meth-

ods from the results on artificial problems, especially when the training data is sparse, and on real-world data sets of mpg, abalone and cputime. The better performance of the proposed methods over $\Delta\|\omega\|^2$-RFE, RMB-RFE and SpanB-RFE is interesting and deserves more attentions, since all of them are wrapper-based feature selection methods for SVR. The better performance of the proposed methods over them are probably attributed to the following two differences. Firstly, different ranking criteria are used. The proposed method uses the "aggregate" sensitivity of SVR probabilistic predictions with respect to a feature over the feature space while $\Delta\|\omega\|^2$-RFE uses the sensitivity of the cost function of SVR with respect to a feature and RMB-RFE and SpanB-RFE uses the sensitivity of the error bound of SVR with respect to a feature. Secondly, $\Delta\|\omega\|^2$-RFE, RMB-RFE and SpanB-RFE assume that the SVR solution remains unchanged when a feature is removed within each RFE iteration. This appears to be a strong assumption, judging from the relative performances of RMB-RFE, SpanB-RFE, RMB-RFE* and SpanB-RFE*.

Another advantage of the proposed method is the modest computational load. As mentioned in Section 3, the evaluation of scores for d features includes a one-time training of SVR of about $O(N^{2.3})$ [63] complexity, one-time evaluation of σ^L (or σ^G) of $O(mN)$ where $N = |\mathscr{D}|$, m is the number of support vectors, d-time RP process of $O(dN)$, d-time evaluation of $\sigma^L_{(j)}$ (or $\sigma^G_{(j)}$) of $O(dmN)$, and d-time evaluation of D_{KL} of $O(dN)$. Hence, after one-time training SVR, the proposed criterion scales linearly with respect to d and N. Obviously, $\Delta\|\omega\|^2$-RFE, RMB-RFE and SpanB-RFE have similar computational cost like the proposed methods. However, RMB-RFE* and SpanB-RFE* require

the training of SVR $d-1$ times more than the proposed methods when evaluating the scores for d features. This additional computational load is of $O(dN^{2.3})$, which is significant when N is large.

4.5 Summary

This chapter presents a new wrapper-based feature selection method for SVR. This method measures the importance of a feature by the aggregation, over the feature space, of the sensitivity of SVR probabilistic prediction with and without the feature. Two approximations of the criterion with random permutation process are proposed. The numerical experiment on both artificial and real-world problems suggests that the proposed method generally performs as least as well, if not better than three benchmark methods. The advantage of the proposed methods is more significant when the training data is sparse, or has a low samples-to-features ratio. As a wrapper method, the computational cost of proposed methods is moderate.

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\|\omega\|\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
1	16.47	16.47	1.00	16.86	0.75	22.45	**0.00+**	16.47	1.00	22.45	**0.00+**	31.79	**0.00+**	22.21	**0.00+**	16.97	0.69
2	7.71	7.71	1.00	16.32	**0.00+**	18.06	**0.00+**	7.71	1.00	17.77	**0.00+**	18.35	**0.00+**	17.75	**0.00+**	8.59	0.25
3	6.76	6.76	1.00	15.51	**0.00+**	15.67	**0.00+**	7.54	0.22	17.39	**0.00+**	16.29	**0.00+**	17.31	**0.00+**	7.69	0.15
4	6.81	6.81	1.00	13.46	**0.00+**	13.46	**0.00+**	6.88	0.91	15.71	**0.00+**	14.30	**0.00+**	15.96	**0.00+**	7.30	0.41
5	6.82	6.82	1.00	11.84	**0.00+**	9.79	**0.00+**	6.71	0.86	13.62	**0.00+**	13.51	**0.00+**	13.96	**0.00+**	6.65	0.78
6	6.68	6.70	0.98	6.68	1.00	6.44	0.67	6.63	0.92	11.16	**0.00+**	8.62	**0.04+**	11.17	**0.00+**	6.50	0.63
7	6.20	6.20	1.00	6.20	1.00	6.20	1.00	6.20	1.00	6.20	1.00	6.20	1.00	6.20	1.00	6.20	1.00
SCC measure																	
1	0.73	0.73	1.00	0.72	0.75	0.63	**0.00+**	0.73	1.00	0.63	**0.00+**	0.48	**0.00+**	0.63	**0.00+**	0.72	0.69
2	0.87	0.87	1.00	0.73	**0.00+**	0.70	**0.00+**	0.87	1.00	0.70	**0.00+**	0.69	**0.00+**	0.71	**0.00+**	0.86	0.25
3	0.89	0.89	1.00	0.74	**0.00+**	0.74	**0.00+**	0.88	0.22	0.71	**0.00+**	0.73	**0.00+**	0.71	**0.00+**	0.87	0.15
4	0.89	0.89	1.00	0.78	**0.00+**	0.78	**0.00+**	0.89	0.91	0.74	**0.00+**	0.76	**0.00+**	0.74	**0.00+**	0.88	0.41
5	0.89	0.89	1.00	0.81	**0.00+**	0.84	**0.00+**	0.89	0.86	0.78	**0.00+**	0.78	**0.00+**	0.86	**0.00+**	0.89	0.78
6	0.89	0.89	0.98	0.89	1.00	0.90	0.67	0.89	0.92	0.82	**0.00+**	0.86	**0.04+**	0.82	**0.00+**	0.90	0.63
7	0.90	0.89	1.00	0.90	1.00	0.89	1.00	0.89	1.00	0.90	1.00	0.90	1.00	0.90	1.00	0.90	1.00

Table 4.3: *t*-test on mpg data set.

SINGAPORE

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\omega\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
1	6.73	6.67	0.63	6.10	**0.00-**	6.15	**0.00-**	6.27	**0.00-**	7.15	**0.00+**	6.97	**0.01+**	7.12	**0.00+**	6.18	**0.00-**
2	4.95	4.95	0.95	6.02	**0.00+**	5.90	**0.00+**	4.97	0.51	6.37	**0.00+**	6.82	**0.00+**	6.67	**0.00+**	4.95	0.92
3	4.74	4.74	1.00	5.39	**0.00+**	5.62	**0.00+**	4.80	0.05	5.16	**0.00+**	6.29	**0.00+**	5.96	**0.00+**	4.87	**0.00+**
4	4.69	4.69	0.99	5.39	**0.00+**	5.41	**0.00+**	4.72	0.42	4,83	**0.00+**	5.87	**0.00+**	5.73	**0.00+**	4.79	**0.00+**
5	4.67	4.67	0.95	5.34	**0.00+**	5.29	**0.00+**	4.66	0.88	4.73	0.17	5.29	**0.00+**	5.28	**0.00+**	4.76	**0.01+**
6	4.64	4.64	0.87	5.21	**0.00+**	5.28	**0.00+**	4.63	0.67	4.71	0.16	4.89	**0.00+**	4.88	**0.00+**	4.70	0.06
7	4.62	4.62	0.98	4.59	0.32	4.90	**0.00+**	4.60	0.62	4.63	0.78	4.71	0.07	4.63	0.79	4.67	0.12
8	4.57	4.57	1.00	4.58	1.00	4.57	1.00	4.57	1.00	4.58	1.00	4.58	1.00	4.58	1.00	4.58	1.00
SCC measure																	
1	0.36	0.36	0.63	0.42	**0.00-**	0.41	**0.00-**	0.40	**0.00-**	0.32	**0.00+**	0.33	**0.01+**	0.32	**0.00+**	0.41	**0.00-**
2	0.53	0.53	0.95	0.42	**0.00+**	0.44	**0.00+**	0.53	0.51	0.39	**0.00+**	0.35	**0.00+**	0.36	**0.00+**	0.53	0.92
3	0.55	0.55	1.00	0.49	**0.00+**	0.46	**0.00+**	0.54	0.05	0.51	**0.00+**	0.40	**0.00+**	0.43	**0.00+**	0.54	**0.00+**
4	0.55	0.55	0.99	0.49	**0.00+**	0.48	**0.00+**	0.55	0.42	0.54	**0.02+**	0.44	**0.00+**	0.45	**0.00+**	0.54	**0.00+**
5	0.55	0.56	0.95	0.49	**0.00+**	0.50	**0.00+**	0.56	0.88	0.55	0.17	0.50	**0.00+**	0.50	**0.00+**	0.55	**0.01+**
6	0.56	0.56	0.87	0.50	**0.00+**	0.50	**0.00+**	0.56	0.67	0.55	0.16	0.53	**0.00+**	0.54	**0.00+**	0.55	0.06
7	0.56	0.56	0.98	0.56	0.32	0.53	**0.00+**	0.56	0.62	0.56	0.78	0.55	0.07	0.56	0.78	0.53	0.12
8	0.56	0.56	1.00	0.56	1.00	0.56	1.00	0.56	1.00	0.56	1.00	0.56	1.00	0.56	1.00	0.56	1.00

Table 4.4: *t*-test on abalone data set.

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\omega\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
2	40.39	64.81	**0.00+**	297.51	**0.00+**	293.6	**0.00+**	75.45	**0.00+**	276.56	**0.00+**	141.00	**0.00+**	295.11	**0.00+**	291.26	**0.00+**
4	18.99	19.33	0.55	279.65	**0.00+**	82.44	**0.00+**	60.09	**0.00+**	242.23	**0.00+**	32.66	0.15	222.18	**0.00+**	247.39	**0.00+**
6	19.20	19.22	0.97	116.14	**0.00+**	28.57	0.32	39.89	**0.00+**	167.24	**0.00+**	16.60	0.05	112.87	**0.00+**	206.61	**0.00+**
8	20.66	21.28	0.32	19.69	0.07	20.49	0.78	29.36	**0.00+**	19.96	0.25	17.54	0.06	78.51	**0.00+**	124.44	**0.00+**
10	21.64	22.52	0.24	20.68	0.15	22.49	0.28	25.61	**0.00+**	20.81	0.25	19.67	0.07	55.55	**0.00+**	59.30	**0.00+**
12	23.78	23.78	1.00	23.78	1.00	23.78	1.00	23.78	1.00	23.78	1.00	23.78	1.00	23.78	1.00	23.78	1.00
SCC measure																	
2	0.89	0.82	**0.00+**	0.16	**0.00+**	0.17	**0.00+**	0.79	**0.00+**	0.22	**0.00+**	0.60	**0.00+**	0.17	**0.00+**	0.17	**0.00+**
4	0.95	0.95	0.55	0.21	**0.00+**	0.77	**0.00+**	0.83	**0.00+**	0.31	**0.00+**	0.91	0.15	0.37	**0.00+**	0.29	**0.00+**
6	0.95	0.95	0.97	0.67	**0.00+**	0.92	0.32	0.89	**0.00+**	0.52	**0.00+**	0.95	0.05	0.68	**0.00+**	0.41	**0.00+**
8	0.94	0.94	0.32	0.94	0.07	0.94	0.78	0.92	**0.00+**	0.94	0.25	0.95	0.06	0.78	**0.00+**	0.65	**0.00+**
10	0.94	0.94	0.24	0.94	0.15	0.94	0.28	0.93	**0.00+**	0.94	0.25	0.94	0.07	0.84	**0.00+**	0.84	**0.00+**
12	0.93	0.93	1.00	0.93	1.00	0.93	1.00	0.93	1.00	0.93	1.00	0.93	1.00	0.93	1.00	0.93	1.00

Table 4.5: *t*-test on cputime data set.

SINGAPORE

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\omega\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
2	19.00	19.00	1.00	29.36	**0.00+**	19.00	1.00	28.99	**0.00+**	64.09	**0.00+**	62.60	**0.00+**	46.80	**0.00+**	19.00	1.00
4	16.00	15.94	0.98	25.46	**0.00+**	14.86	0.60	15.19	0.71	38.98	**0.00+**	56.52	**0.00+**	23.22	**0.01+**	13.97	0.35
6	13.74	13.59	0.94	16.28	0.26	13.90	0.94	13.69	0.98	28.96	**0.00+**	50.93	**0.00+**	18.33	**0.03+**	12.63	0.54
8	11.47	12.46	0.54	15.24	0.06	11.54	0.96	12.02	0.74	24.63	**0.00+**	43.99	**0.00+**	11.38	0.95	11.34	0.93
10	9.57	10.76	0.40	11.32	0.18	10.49	0.50	11.08	0.28	12.25	0.07	37.94	**0.00+**	11.71	0.15	11.60	0.18
12	10.12	10.12	1.00	9.45	0.62	9.51	0.65	10.36	0.87	10.81	0.63	17.83	**0.00+**	10.81	0.65	10.69	0.70
13	10.48	10.48	1.00	10.48	1.00	10.48	1.00	10.48	1.00	10.48	1.00	10.48	1.00	10.48	1.00	10.48	1.00
SCC measure																	
2	0.77	0.77	1.00	0.65	**0.00+**	0.77	1.00	0.65	**0.00+**	0.23	**0.00+**	0.25	**0.00+**	0.45	**0.00+**	0.77	1.00
4	0.80	0.80	0.98	0.70	**0.00+**	0.82	0.60	0.81	0.71	0.54	**0.00+**	0.34	**0.00+**	0.73	**0.01+**	0.83	0.35
6	0.83	0.83	0.94	0.80	0.26	0.83	0.94	0.83	0.98	0.66	**0.00+**	0.41	**0.00+**	0.79	**0.03+**	0.84	0.54
8	0.86	0.85	0.54	0.82	0.06	0.86	0.96	0.85	0.74	0.71	**0.00+**	0.49	**0.00+**	0.86	0.95	0.86	0.93
10	0.88	0.87	0.40	0.86	0.18	0.87	0.50	0.86	0.28	0.85	0.07	0.56	**0.00+**	0.86	0.15	0.86	0.18
12	0.88	0.88	1.00	0.88	0.62	0.88	0.65	0.87	0.87	0.86	0.63	0.79	**0.00+**	0.87	0.64	0.86	0.70
13	0.87	0.87	1.00	0.87	1.00	0.87	1.00	0.87	1.00	0.87	1.00	0.87	1.00	0.87	1.00	0.87	1.00

Table 4.6: *t*-test on housing data set.

SINGAPORE

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\omega\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
2	.00022	.00022	0.91	.00017	**0.00-**	.00022	0.91	.00022	0.91	.00021	0.51	.00026	0.08	.00032	**0.00+**	.00018	**0.00-**
4	.00018	.00018	0.93	.00016	0.07	.00025	**0.00+**	.00017	0.19	.00021	0.11	.00023	**0.02+**	.00020	0.28	.00022	**0.04+**
6	.00021	.00021	1.00	.00019	0.08	.00026	**0.00+**	.00020	0.29	.00021	0.88	.00021	0.16	.00019	0.12	.00024	0.06
8	.00020	.00020	0.97	.00023	0.04	.00026	0.05	.00020	0.95	.00022	0.31	.00023	0.09	.00019	0.54	.00025	**0.00+**
10	.00020	.00020	0.99	.00023	0.05	.00025	0.05	.00020	0.95	.00022	0.14	.00023	0.12	.00019	0.78	.00024	**0.01+**
12	.00021	.00021	1.00	.00023	0.16	.00025	0.05	.00020	0.66	.00023	0.27	.00022	0.48	.00020	0.59	.00023	0.19
14	.00021	.00021	1.00	.00021	1.00	.00021	1.00	.00021	1.00	.00021	1.00	.00021	1.00	.00021	1.00	.00021	1.00
SCC measure																	
2	0.89	0.89	0.91	0.95	**0.00-**	0.89	0.91	0.89	0.91	0.52	0.51	0.38	0.08	0.18	**0.00+**	0.79	**0.00+**
4	0.84	0.84	0.93	0.92	0.07	0.83	**0.00+**	0.86	0.19	0.73	0.11	0.46	**0.02+**	0.58	0.28	0.75	**0.04+**
6	0.79	0.79	1.00	0.84	0.08	0.80	**0.00+**	0.81	0.29	0.79	0.88	0.47	0.16	0.80	0.12	0.75	0.06
8	0.80	0.80	0.97	0.79	0.05	0.79	0.05	0.78	0.95	0.79	0.31	0.48	0.09	0.78	0.54	0.73	**0.00+**
10	0.75	0.75	0.99	0.76	0.05	0.77	0.05	0.76	0.95	0.78	0.14	0.53	0.12	0.76	0.78	0.73	**0.01+**
12	0.74	0.74	1.00	0.73	0.16	0.76	0.05	0.75	0.66	0.76	0.27	0.57	0.48	0.75	0.59	0.75	0.19
14	0.73	0.73	1.00	0.73	1.00	0.73	1.00	0.73	1.00	0.73	1.00	0.73	1.00	0.73	1.00	0.73	1.00

Table 4.7: *t*-test on pyrim data set.

N	SD-L-RFE	SD-G-RFE		mRMR		HSIC-RFE		$\Delta\|\omega\|^2$-RFE		RMB-RFE		SpanB-RFE		RMB-RFE*		SpanB-RFE*	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
MSE measure																	
1	0.020	0.020	1.00	0.020	0.95	0.021	0.95	0.021	0.69	0.021	0.65	0.021	0.65	0.021	0.65	0.021	0.85
10	0.018	0.017	0.92	0.017	0.84	0.019	0.63	0.018	0.80	0.020	0.25	0.021	0.18	0.020	0.20	0.018	0.89
20	0.017	0.017	0.98	0.018	0.75	0.017	0.89	0.017	0.87	0.020	0.15	0.021	0.11	0.020	0.14	0.017	0.93
30	0.017	0.018	0.83	0.018	0.63	0.017	0.94	0.017	0.95	0.019	0.30	0.020	0.17	0.020	0.23	0.018	0.97
40	0.018	0.018	0.94	0.018	0.98	0.018	0.75	0.017	0.85	0.018	0.83	0.019	0.43	0.019	0.46	0.018	0.94
50	0.018	0.018	0.99	0.018	0.91	0.020	0.52	0.018	0.93	0.018	0.93	0.019	0.73	0.019	0.72	0.018	0.96
60	0.018	0.018	1.00	0.018	1.00	0.018	1.00	0.018	1.00	0.018	1.00	0.018	1.00	0.018	1.00	0.018	1.00
SCC measure																	
1	0.12	0.12	1.00	0.08	0.95	0.08	0.95	0.07	0.69	0.094	0.65	0.11	0.85	0.094	0.65	0.11	0.85
10	0.26	0.27	0.92	0.25	0.84	0.19	0.63	0.22	0.80	0.11	0.25	0.11	0.18	0.12	0.20	0.26	0.89
20	0.28	0.29	0.98	0.22	0.75	0.26	0.89	0.28	0.87	0.12	0.15	0.12	0.11	0.14	0.14	0.30	0.93
30	0.29	0.26	0.83	0.20	0.62	0.26	0.94	0.29	0.95	0.18	0.30	0.14	0.17	0.17	0.23	0.28	0.97
40	0.26	0.26	0.94	0.26	0.98	0.22	0.75	0.27	0.85	0.25	0.83	0.17	0.43	0.17	0.46	0.27	0.94
50	0.25	0.25	0.99	0.26	0.90	0.17	0.52	0.26	0.93	0.22	0.94	0.19	0.73	0.21	0.72	0.26	0.96
60	0.25	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00	0.25	1.00

Table 4.8: *t*-test on triazines data set.

SINGAPORE

Chapter 5

Feature Selection via Mutual Information Estimation

This chapter proposes a new feature selection method using a mutual information based criterion that measures the importance of a feature in a backward selection framework. It considers the dependency among many features and uses either one of two well known probability density function estimation methods when computing the criterion. The proposed approach is compared with existing mutual information based methods and another sophisticated filter method on many artificial and real world problems. The numerical results show that the proposed method can effectively identify the important features in data sets having dependency among many features and is at least as good as, if not better than, the benchmark methods.

This chapter is organized as follow: Section 5.1 review two well-known density esti-

mation methods. Detailed accounts of the proposed feature selection criterion are presented in Section 5.2. Some connections between the proposed method and some other methods are built in Section 5.3. Section 5.4 reports extensive numerical studies of the proposed method in comparison to some existing methods in the literature, followed by the summary in Section 5.5.

5.1 Preliminary

As shown in 2.1.3 of Chapter 2, entropy and mutual information rely on the values of (conditional) density functions. This section reviews two commonly used probability density estimation methods.

Parzen Window (PW) [60, 20] is a well-known density estimation method that has been widely used in various applications. Given a data set $\{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$, PW provides the estimate of probability density of x, $p(x)$, in the form of

$$\hat{p}(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i), \tag{5.1}$$

where $\alpha_i = \frac{1}{N}$, $\forall i = 1, .., N$, is a weighting coefficient and $K(x, x_i)$ is an appropriate window function typically chosen as the Gaussian function, $\frac{1}{(\sqrt{2\pi}\sigma)^d} \exp(-\frac{\|x-x_i\|^2}{2\sigma^2})$, with hyperparameter σ. Determination of σ is often done by minimizing an appropriate negative log-likelihood function [20]. Specifically, given $\hat{p}(x)$, the likelihood function,

as a function of σ, of observing data set $\{x_i\}_{i=1}^N$ is given by

$$L(\sigma) = \Pi_{j=1}^N p(x_j;\sigma) = \Pi_{j=1}^N \sum_{i=1}^N \alpha_i K(x_j, x_i;\sigma)$$

and the corresponding negative log-likelihood function becomes

$$Ln(\sigma) = -\log(\Pi_{j=1}^N p(x_j;\sigma)) = -\sum_{j=1}^N \sum_{i=1}^N \alpha_i K(x_j, x_i;\sigma).$$

Remark 5.1.1. *If σ is given, it is easy to see that the evaluation of $\hat{p}(x)$ for one x using (5.1) is $O(N)$, or $O(NM)$ for M values of x.*

If $M >> N$, it is possible to lower the computational cost. Girolami et al. [27] proposed a sparse PW method, called Reduced Set Density Estimation (RSDE), that uses the same expression of (5.1) but with $\{\alpha_i\}_{i=1}^N$ determined by the solution of the following optimization problem:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^N \sum_{j=1}^N \left[\frac{1}{2}\alpha_i\alpha_j \tilde{K}(x_i, x_j) - \frac{1}{N}\alpha_i K(x_i, x_j)\right] \\ s.t \quad & \sum_i \alpha_i = 1, \quad \alpha_i \geq 0, \quad i = 1, \cdots, N \end{aligned} \tag{5.2}$$

with $\tilde{K}(x, x_i) = \frac{1}{(2\sqrt{\pi}\sigma)^d} \exp(-\frac{\|x-x_i\|^2}{4\sigma^2})$.

The quadratic optimization problem of (5.2) is derived from minimizing the integrated squared error between $p(x)$ and $\hat{p}(x)$. One advantage of RSDE is that the solution of (5.2) is sparse with only a few non-zero α_i.

Remark 5.1.2. *The numerical solution of (5.2) using sequential minimal optimization*

method [62] has a computational complexity of about $O(N^2)$. Suppose the solution of (5.2) contains $\tilde{N}(< N)$ non-zero α_i. The evaluation of (5.1) for M values of x requires $O(\tilde{N}M)$. Hence, the total complexity is $O(N^2) + O(\tilde{N}M)$ using the RSDE approach. If $M >> N > \tilde{N}$, the RSDE approach can be more efficient than PW.

5.2 The Proposed Method

The proposed feature selection method is for *c*-class classification problem. Recalling the mutual information method proposed by Battiti [4] and Kwak et al. [46] as reviewed in Section 2.2.1, we use the similar idea but in a backward feature selection framework. The backward selection framework is implemented in an iterative loop and starts with the full feature set, $\mathscr{I}_0 = \mathscr{I}$. It eliminates the least important feature in $\mathscr{D}$ from the set of remaining features at every iteration and has the advantage that interactions among all remaining features are considered. Let $z \in \mathbb{R}^v$ be a vector obtained by taking v of the d features from $x \in \mathbb{R}^d$ and $z_{-j} \in \mathbb{R}^{v-1}$ be the vector obtained from z with the j^{th} feature removed. The proposed criterion is

$$S(j) = I(z_{-j}; y). \tag{5.3}$$

Since $I(z_{-j}; y)$ measures the dependency of z_{-j} and y, the removal of a non-important j feature from z will increase its value. Hence, the j that maximizes $S(j)$ over $j \in \mathscr{I}_\ell$ is the least important feature. Here, $\mathscr{I}_\ell$ is the set of remaining features at iteration ℓ.

Criterion (5.3) is also equivalent to

$$S_1(j) = I(z;y) - I(z_{-j};y) \tag{5.4}$$

but with the intention of looking for the minimizing j over all $j \in \mathscr{I}_\ell$. This equivalence is clear since $I(z;y)$ is a constant in a fixed iteration and $\max_j I(z_{-j};y) = \min_j -I(z_{-j};y)$. When written in full, (5.3) or (5.4) becomes

$$\begin{aligned} S(j) =& I(z_{-j};y) = \int\int p(z_{-j},y) \log \frac{p(z_{-j},y)}{p(z_{-j})p(y)} dz_{-j} dy \\ =& \mathbb{E}_{-z,y}\left[\log \frac{p(z_{-j},y)}{p(z_{-j})p(y)}\right] \approx \frac{1}{N}\sum_{i=1}^{N} \log \frac{p(z_{-j,i},y_i)}{p(z_{-j,i})p(y_i)}. \end{aligned} \tag{5.5}$$

The notation $(z_{-j,i},y_i)$ refers to the sample obtained from the i^{th} sample, (x_i,y_i), of $\mathscr{D}$. The expression of (5.5) is not the most ideal for computations. It contains two density functions, $p(z_{-j},y)$ and $p(z_{-j})$, that have to be estimated for every $j \in \mathscr{I}_\ell$. Their estimations using PW or RSDE have complexity of $O(2N^2|\mathscr{I}_\ell|)$ and $(O(2N^2|\mathscr{I}_\ell|) + O(2\tilde{N}N|\mathscr{I}_\ell|))$ respectively, following *Remarks* 5.1.1 and 5.1.2.

Further simplification of (5.5) is possible for computational expediency. Recall that a sample $x_i \in \omega_k$ (or $z_i \in \omega_k$) if and only if $y_i = k$. Let $\mathscr{D}$ at iteration ℓ be decomposed into $\mathscr{D}^k_{-j} = \{(z_{-j,i},y_i)|y_i = k\}$ for $k = 1,\cdots,c$ and for every $j \in \mathscr{I}_\ell$ with $|\mathscr{D}^k_{-j}| = N_k$.

Expression (5.5) can be simplified to:

$$\begin{aligned} S(j) =& \mathbb{E}_{-z,y}\left[\log\frac{p(z_{-j}|y)}{p(z_{-j})}\right] = \mathbb{E}_{-z,y}\left[\log\frac{p(z_{-j}|y)}{\sum_{\tilde{k}=1}^{c} p(z_{-j}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\right] \\ \approx& \frac{1}{N}\sum_{i=1}^{N}\log\frac{p(z_{-j,i}|y_i)}{\sum_{\tilde{k}=1}^{c} p(z_{-j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})} \\ =& \frac{1}{N}\sum_{k=1}^{c}\sum_{i:y_i=k}\log\frac{p(z_{-j,i}|\omega_k)}{\sum_{\tilde{k}=1}^{c} p(z_{-j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})} \\ \approx& \frac{1}{N}\sum_{k=1}^{c}\sum_{i:y_i=k}\log\frac{\hat{p}(z_{-j,i}|\omega_k)}{\sum_{\tilde{k}=1}^{c} \hat{p}(z_{-j,i}|\omega_{\tilde{k}})\hat{P}(\omega_{\tilde{k}})} \end{aligned} \tag{5.6}$$

where $P(\omega_{\tilde{k}})$ is the prior probability of class $\omega_{\tilde{k}}$ which can be estimated using $\hat{P}(\omega_{\tilde{k}}) = \frac{N_{\tilde{k}}}{N}$.

Consider the numerical evaluation of (5.6) using PW. Following *Remark* 5.1.1, the evaluation of $\sum_{k=1}^{c}\sum_{i:y_i=k}\sum_{\tilde{k}=1}^{c}\hat{p}(z_{-j,i}|\omega_{\tilde{k}}) = \sum_{i=1}^{N}\sum_{\tilde{k}=1}^{c}\hat{p}(z_{-j,i}|\omega_{\tilde{k}})$ requires $O(N^2)$ operations for one choice of j. Here, the standard assumption [20] is adopted in that a sample $z_{-j,i}$ is used to estimate $p(z_{-j}|\omega_k)$ only when $y_i = k$. For all $j \in \mathscr{I}_\ell$, the evaluation $S(j)$ has the complexity of $O(N^2|\mathscr{I}_\ell|)$. This suggests that evaluation of $S(j)$ via (5.6) is about half the computational cost needed via (5.5).

Consider the approach of RSDE. Equation (5.6) requires expression of $\hat{p}(z_{-j}|\omega_{\tilde{k}})$ for all $\tilde{k} = 1,\cdots,c$. Following *Remark* 5.1.2, this means that (5.2) has to be solved c times, each time for one k and using $\mathscr{D}_{-j}^{k}$. The evaluation of $\sum_{k=1}^{c}\sum_{i:y_i=k}\sum_{\tilde{k}=1}^{c}\hat{p}(z_{-j,i}|\omega_{\tilde{k}})$ requires $O(c\tilde{N}_{\tilde{k}}N)$ operations for one choice of j. Hence, the evaluation of $S(j)$ for all $j \in \mathscr{I}_\ell$ has the complexity of $O(cN_{k_{\max}}^2|\mathscr{I}_\ell|) + O(c\tilde{N}_{\max}N|\mathscr{I}_\ell|)$ where $N_{k_{\max}} = \max\{N_k : k = 1,\cdots,c\}$ and $\tilde{N}_{\max} = \max\{\tilde{N}_k : k = 1,\cdots,c\}$.

The solution of (5.6) using RSDE can be further simplified to avoid solving (5.2) $|\mathscr{I}_\ell|$ times at each iteration ℓ. This is made possible using a random permutation (RP) procedure as mentioned in Section 3.2 of Chapter 3. Let $x_{(j)} \in \mathbb{R}^d$ denote the sample derived from x after this RP process on the j^{th} feature. The following result is known.

Theorem 5.2.1. *Assume that $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^N$ is sufficiently rich, then*

$$\hat{p}(z_{-j}|\omega_k) = \hat{p}(z_{(j)}|\omega_k) \tag{5.7}$$

for any $k = 1, \cdots, c$.

The assumption of $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^N$ being sufficiently rich is needed to ensure that the RP process destroys any correlation between the j^{th} feature and all other features in $\mathscr{D}$. While this assumption may not be easy to verify, $\hat{p}(z_{(j)}|\omega_k)$ is an excellent approximation to $\hat{p}(z_{-j}|\omega_k)$ for all data sets in our experiments.

The use of Theorem 5.2.1 to simplify the RSDE computations of (5.6) is now possible. The conditional density function $\hat{p}(z_{-j}|\omega_k)$ in (5.6) is replaced by $\hat{p}(z_{(j)}|\omega_k)$ for all $j \in \mathscr{I}_\ell$. This means that (5.2) need not be solved $|\mathscr{I}_\ell|$ times, each with $\mathscr{D}^k_{-j}$ for a different j. Instead, it is solved once for $\hat{p}(z|\omega_k)$ using $\mathscr{D}^k := \{(z_i, y_i)|y_i = k\}$. Thereafter, $\hat{p}(z_{(j)}|\omega_k)$ is obtained from $\hat{p}(z|\omega_k)$ following the RP procedure for every $j \in \mathscr{I}_\ell$. Correspondingly, (5.6) becomes

$$S(j) \approx \frac{1}{N} \sum_{k=1}^{c} \sum_{i:y_i=k} \log \frac{\hat{p}(z_{(j),i}|\omega_k)}{\sum_{\tilde{k}=1}^{c} \hat{p}(z_{(j),i}|\omega_{\tilde{k}}) \hat{P}(\omega_{\tilde{k}})}. \tag{5.8}$$

As a result of this simplification, the complexity associated with the solution of (5.2) drops from $O(cN_{k_{\max}}^2|\mathscr{I}_\ell|)$ to $O(cN_{k_{\max}}^2)$ at each iteration. The overall complexity for the evaluation of (5.6) using RSDE becomes $O(cN_{k_{\max}}^2)+O(c\tilde{N}_{\max}N|\mathscr{I}_\ell|)$.

Henceforth, two proposed criteria (5.6) and (5.8) used in backward framework are denoted as methods MI-PW and MI-RSDE, respectively.

5.3 Connection with Other Methods

The connection between the proposed approach and the method by Kwak et at. [46] is made clear in this section. As mentioned before, criterion (2.28) is used in a forward feature selection framework in [46]. Hence, the obvious difference is in the choice of the selection framework. A less obvious difference is the way in which the criterion is computed. This difference is best described using both methods for criterion (2.28). Using Kwak's method of [46], (2.28) is evaluated by

$$I(z_{+j};y) = -\mathbb{E}_y\left[\log p(y)\right] + \mathbb{E}_{z_{+j},y}\left[\log p(y|z_{+j})\right] \tag{5.9}$$

$$= -\int p(y)\log p(y)dy + \int\int p(z_{+j},y)\log p(y|z_{+j})dz_{+j}dy. \tag{5.10}$$

The first term of the last equation is independent of j and its computations are not relevant. The second term is further expanded as

$$\int\int p(z_{+j})p(y|z_{+j})\log p(y|z_{+j})dz_{+j}dy \tag{5.11}$$

$$=\int p(z_{+j})\left[\int p(y|z_{+j})\log p(y|z_{+j})dy\right]dz_{+j} \tag{5.12}$$

$$=\int p(z_{+j})\left[\sum_{k=1}^{c} P(\omega_k|z_{+j})\log P(\omega_k|z_{+j})\right]dz_{+j} \tag{5.13}$$

$$\approx\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{c}\left[P(\omega_k|z_{+j,i})\log P(\omega_k|z_{+j,i})\right] \tag{5.14}$$

$$=\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{c}\left[\frac{p(z_{+j,i}|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c} p(z_{+j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\log\frac{p(z_{+j,i}|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c} p(z_{+j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\right]. \tag{5.15}$$

As shown in (5.10), the second term of the last equation (5.10) is $\mathbb{E}_{z_{+j},y}\left[\log p(y|z_{+j})\right]$. The approach by Kwak et al. replaces $p(z_{+j},y)$ by $p(z_{+j})p(y|z_{+j})$ in (5.11). This effectively replaces $\mathbb{E}_{z_{+j},y}\left[\log p(y|z_{+j})\right]$ by $\mathbb{E}_{z_{+j}}\left[\sum_{k=1}^{c} P(\omega_k|z_{+j})\log P(\omega_k|z_{+j})\right]$, as shown in (5.13). This change implies that Kwak's approach assumes samples x_i's are independent and identically distributed (i.i.d.).

In contrast, the proposed method assumes that data points (x_i,y_i) in $\mathscr{D}$ are independent and identically distributed samples, as shown in (5.5). Using the proposed approach, the second term of (5.10) would be expanded as

$$\mathbb{E}_{z_{+j},y}\left[\log p(y|z_{+j})\right]=\mathbb{E}_{z_{+j},y}\left[\log\frac{p(z_{+j}|y)p(y)}{\sum_{\tilde{k}=1}^{c} p(z_{+j}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\right] \tag{5.16}$$

$$\approx\frac{1}{N}\sum_{i=1}^{N}\log\frac{p(z_{+j,i}|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c} p(z_{+j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})} \tag{5.17}$$

$$=\frac{1}{N}\sum_{k=1}^{c}\sum_{i:y_i=k}\log\frac{p(z_{+j,i}|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c} p(z_{+j,i}|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}. \tag{5.18}$$

Clearly, (5.18) is different from (5.15).

Remark 5.3.1. *Theoretically, both i.i.d. assumptions are correct. However, as shown the following derivation from expression (5.19) to (5.22), the assumption that* $(x_i,y_i)'s$ *are i.i.d. needs less approximations than the assumption that* $x_i's$ *are i.i.d.. It is known that less approximation used less error incurred.*

(x_i,y_i)***'s are i.i.d.***

$$\begin{aligned} I(x;y) &= \int\int p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy \\ &= \frac{1}{N}\sum_{i=1}^{N}\log\frac{p(x_i,y_i)}{p(x_i)p(y_i)} \\ &= \frac{1}{N}\sum_{i=1}^{N}\log\frac{p(x_i|y_i)}{p(x_i)} \end{aligned} \tag{5.19}$$

x_i***'s are i.i.d.***

$$\begin{aligned} I(x;y) &= \int\int p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy \\ &= \int p(x)\int p(y|x)\log\frac{p(x,y)}{p(x)p(y)}dxdy \\ &= \frac{1}{N}\sum_{i=1}^{N}\int p(y|x_i)\log\frac{p(x_i,y)}{p(x_i)p(y)}dy \\ &= \frac{1}{N}\sum_{i=1}^{N}\int \frac{p(x_i|y)p(y)}{p(x_i)}\log\frac{p(x_i|y)}{p(x_i)}dy \end{aligned} \tag{5.20}$$

Estimation

Expression (5.19) is

$$\frac{1}{N}\sum_{i=1}^{N}\log\frac{p(x_i|y_i)}{p(x_i)}=\frac{1}{N}\sum_{k=1}^{c}\sum_{i;y_i=k}\log\frac{p(x_{(j),i}|\omega_k)}{\sum_{\tilde{k}=1}^{c}p(x_{(j),i}|\omega_{\tilde{k}})\hat{P}(\omega_{\tilde{k}})} \tag{5.21}$$

Expression (5.20) is

$$\begin{aligned}&\frac{1}{N}\sum_{i=1}^{N}\int\frac{p(x_i|y)p(y)}{p(x_i)}\log\frac{p(x_i|y)}{p(x_i)}dy\\=&\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{c}\left[\frac{p(x_i|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c}p(x_i|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\log\frac{p(x_i|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c}p(x_i|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}\right].\end{aligned} \tag{5.22}$$

Compared to expression (5.21), expression (5.22) needs additional approximation on the term $\frac{p(x_i|\omega_k)P(\omega_k)}{\sum_{\tilde{k}=1}^{c}p(x_i|\omega_{\tilde{k}})P(\omega_{\tilde{k}})}$*. Extremely, we can further use the trick of Bayes theorems in expression (5.20) and yield the following expression (5.23). In this extreme case, more approximations on a series of probabilistic terms, namely* $p(x_i^2|x_i^1)p(x_i^3|x_i^{1,2})\cdots p(x_i^d|x_i^{1,\cdots,d-1})p(y_i|x_i^d)$*, are demanded.*

$$\begin{aligned}&I(x;y)\\=&\int\int p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy\\=&\int\cdots\int p(x^1)p(x^2|x^1)p(x^3|x^{1,2})\cdots p(x^d|x^{1,\cdots,d-1})p(y|x^d)\log\frac{p(x,y)}{p(x)p(y)}dx^1\cdots dx^d dy\\=&\frac{1}{N}\sum_{i=1}^{N}\int\cdots\int p(x_i^2|x_i^1)p(x_i^3|x_i^{1,2})\cdots p(x_i^d|x_i^{1,\cdots,d-1})p(y_i|x_i^d)\log\frac{p(x_i,y_i)}{p(x_i)p(y_i)}dx^2\cdots dx^d dy\end{aligned} \tag{5.23}$$

5.4 Numerical Experiment

Numerical experiments of MI-PW and MI-RSDE and three benchmark methods are conducted on artificial and real-world data sets. The experiment is done in Matlab 2009a on Window Vista PC with 3 GHz of Intel Core 2 processor E8400 and 8GB of RAM. The benchmark methods include two existing mutual information based feature selection methods, mRMR [53] of (2.31) and Kwak [46] of (2.28), and Dependence Maximization method (HSIC) [74]. The mRMR is used as a representative method of those stated by (2.29)-(2.34) since it has similar performance to them on data having three or more interacting features. Following [53, 46, 74] , mRMR and Kwak are used in the forward selection framework with HSIC backward. Density functions in (2.31) for mRMR and (2.28) for Kwak are estimated using histograms and PW respectively. To investigate the effect of sparsity of the training data, decreasing sizes of $|\mathscr{D}_{trn}|$ are used. As done in Chapters 3 and 4, paired t-test between MI-PW and each of the other methods is conducted using different number of top ranked features.

Support vector machine (SVM) with Gaussian kernel $G(x_i,x_j) = \exp(-\kappa||x_i - x_j||^2)$ is used as the classifier for performance evaluation of the various selection methods. Training and testing SVM are implemented using the LIBSVM package [10]. In each experiment, the hyper-parameter σ in PW and RSDE are chosen by a 5-fold cross-validation for the each realization of $\mathscr{D}_{trn}$, and the parameter corresponding to the smallest negative log-likelihood function value is chosen. Kernel parameter κ and regularized parameter C are chosen by 5-fold cross-validation on first five realizations of $\mathscr{D}_{trn}$, and the parameter corresponding to the lowest average error rate is chosen. The grid over (σ,κ,C) is

$[2^{-3}, 2^{-2.5}, \cdots, 2^{3}] \times [2^{-6}, 2^{-5}, \cdots, 2^{5}] \times [2^{-3}, 2^{-2}, \cdots, 2^{6}]$.

5.4.1 Artificial Data Sets

Monk Data Sets

Monk data sets [1] include 3 problems (Monk-1 Monk-2 and Monk-3) as shown in Table 5.1. Each problem has 6 features and relevant features are known according to the given target concepts [1]. Four settings of decreasing $|\mathscr{D}_{trn}|$ at 432, 200, 100 and 50 are considered in experiments.

Table 5.2 presents the number of realizations (out of 30 realizations) that features 1,2,5 in Monk-1 are ranked as the first three most important features by the various methods for the four settings of $|\mathscr{D}_{trn}|$. The advantage of MI-PW over other benchmark methods is evident when the feature selection becomes more challenging with decreasing sizes of the training set. In fact, except MI-PW, none of methods can consistently rank features correctly. Method MI-RSDE also performs better than other benchmark methods and is very effective till $|\mathscr{D}_{trn}|$ reaches 50.

Figure 5.1 shows the plots of average test error rate against the number top-ranked features using all feature selection methods for problem Monk-1. This figure again shows that MI-PW method outperforms other benchmark methods in all different settings of $|\mathscr{D}_{trn}|$. Given top three features, the margins of MI-PW over Kwak, HSIC and mRMR are significant in all settings, and this is confirmed by aforementioned paired t-test. Fig-

[1] As the provided $\mathscr{D}_{trn}$ has too few data, it is exchanged with $\mathscr{D}_{tst}$

	$\|D_{trn}\|$	$\|D_{tst}\|$	d	m	n_r	Target Concept
Monk-1	432	124	6	5	1	$(x_1 = x_2)$ or $(x_5 = 1)$ for Class 1, otherwise Class -1
Monk-2	432	169	6	9	1	Exactly two of $\{x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1, x_5 = 1, x_6 = 1\}$ for Class 1, otherwise Class -1
Monk-3	432	122	6	2	1	$(x_5 = 3$ and $x_4 = 1)$ or $(x_5 \neq 4$ and $x_2 \neq 3)$ for Class 1, otherwise Class -1

Table 5.1: Description of Monk data sets

ure 5.1 also shows that both proposed methods consistently yield curves having one minimal point, at the value where top three features are selected. This is not so for the benchmark methods. Experimental results with Monk-3 data set is given in Table 5.3 and Figure 5.2. They show similar patterns to that of Monk-1: basically, MI-PW shows the best results among all data set settings. Results of Monk-2 are not shown because all features are important as shown in Table 5.1.

Method$\backslash\|\mathscr{D}_{trn}\|$	432	200	100	50
MI-PW	**30**	**30**	**30**	**30**
MI-RSDE	**30**	**30**	**30**	28
Kwak	3	4	14	3
HSIC	29	21	19	9
mRMR	25	28	29	27

Table 5.2: The number of realizations that feature $1, 2, 5$ are successfully ranked in the top three positions over 30 realizations for Monk-1 problem. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold.

Method$\backslash\|\mathscr{D}_{trn}\|$	432	200	100	50
MI-PW	**30**	**30**	**29**	**21**
MI-RSDE	**30**	25	16	14
Kwak	0	0	0	0
HSIC	**30**	29	26	13
mRMR	15	0	2	0

Table 5.3: The number of realizations that feature $2, 4, 5$ are successfully ranked in the top three positions over 30 realizations for Monk-3 problem. The best performance for each $|\mathscr{D}_{trn}|$ is highlighted in bold.

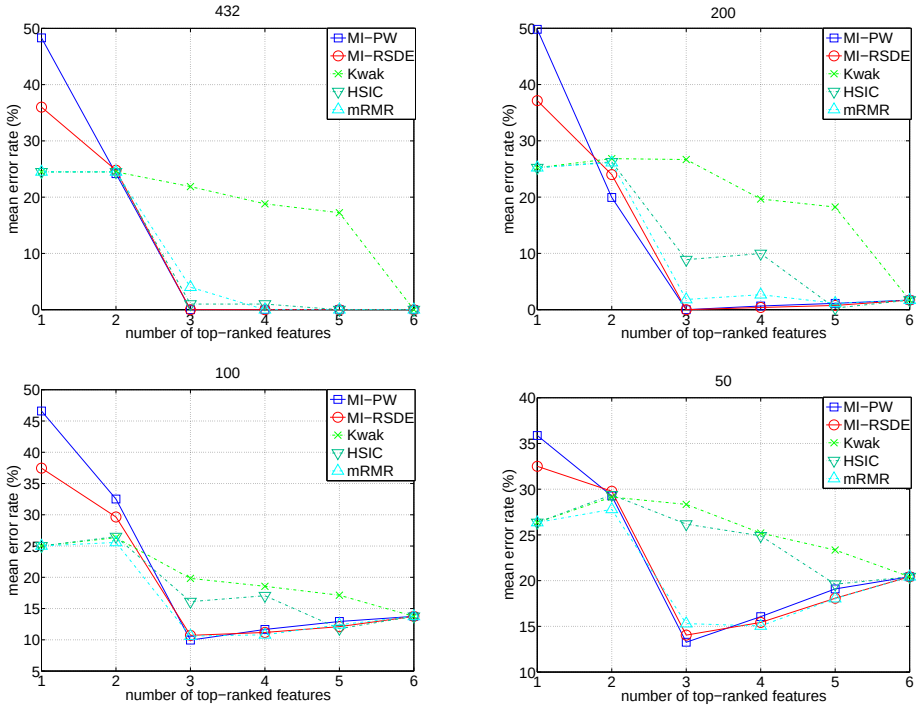


Figure 5.1: Average test error against top-ranked features over 30 realizations of Monk-1 data sets for four training set sizes.

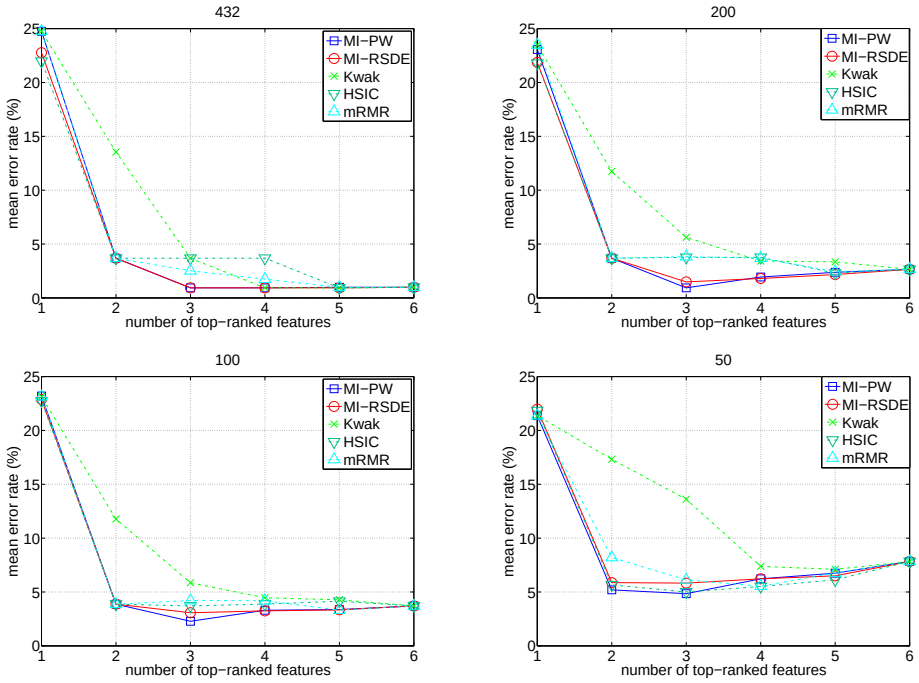


Figure 5.2: Average test error against top-ranked features over 30 realizations of Monk-3 data sets for four training set sizes.

Weston Data Sets

This artificial data set is same as that in Section 3.4.1 of Chapter 3. Four settings with different sizes of the training set ($|\mathscr{D}_{trn}|$=200, 90, 40 and 20) are considered while $|\mathscr{D}_{tst}|$ is maintained at 9800.

Table 5.4 shows the number of realizations (out of 30 realizations) that features 1 and 2 are successfully ranked in the top two positions by the various methods. It is not suprising to note that the backward feature selection methods perform better than the forward methods in all settings. Among the backward selection methods, MI-PW consistently performs best over all four settings and its performance degrades much less than the other two with decreasing $|\mathscr{D}_{trn}|$.

Figure 5.3 again shows that average test error rate against top-ranked features over 30 realizations for all methods except mRMR. Method mRMR is excluded since it fails completely in identifying important features, as shown in Table 5.4. The advantage of MI-PW over other methods is clear, especially for small values of $|\mathscr{D}_{trn}|$. The increase in error rate is less than 4% when $|\mathscr{D}_{trn}|$ decreases from 200 to 20. This is much less than the 13% – 20% exhibited by the other methods.

Method\$\backslash|\mathscr{D}_{trn}|$	200	90	40	20
MI-PW	**30**	**30**	**30**	**26**
MI-RSDE	**30**	**30**	22	12
Kwak	1	20	14	11
HSIC	**30**	**30**	29	18
mRMR	0	0	0	0

Table 5.4: The number of realizations that feature $1,2$ are successfully ranked in the top two positions over 30 realizations for Weston problem.

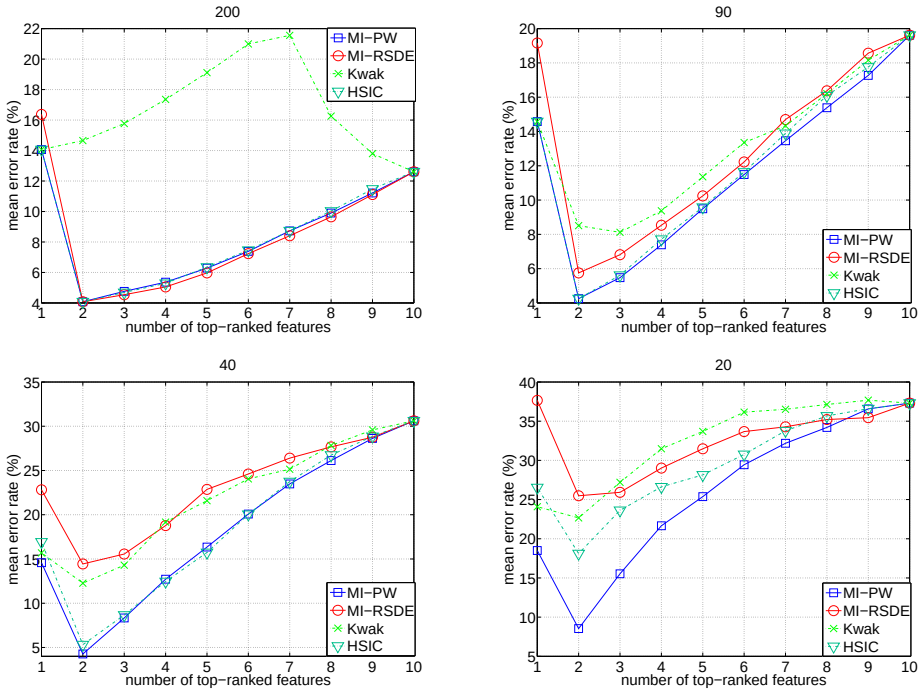


Figure 5.3: Average test error against top-ranked features over 30 realizations of Weston data sets for five training set sizes.

5.4.2 Real Problem

Six real-world data sets from UCI repository [1] are used for evaluation purposes. Description of these data sets and the parameters used in the experiments are given in Table 5.5. The Abalone data set has been transformed into a 3-class classification problem following the procedure by David *et. al.* [16]. Figures 5.4-5.9 show average error rate against the number of top-ranked features for Abalone, WBCD, Glass, Wine, Satimage and Musk respectively. This is followed by the statistical t-test results tabulated in Tables 5.7 to 5.12.

For problem Abalone, Figure 5.4 shows the average test error rate against the number of top-ranked features for both proposed methods and benchmark methods. It can be observed that given the same level of the feature selection (with the same number of

feature selected), MIPW-BW generally yields lower average test error rates than other methods. This is confirmed by the paired t-test's result given in Table 5.7.

For the other real-world problems (WBCD, Glass, Wine, Satimage and Musk), the experimental results show similar patterns to that of Abalone, as shown in Figure 5.5 to Figure 5.9 and Table 5.8 to Table 5.12 respectively. In general, the t-test results show that MIPW-BW performs at least as well, if not better than other methods. There are a few exceptions. For example, The first two rows of Table 5.10 shows that MIPW-BW performs significantly worse than Kwak-FW and HSIC-BW. This should not be seen as a worrying sign since it happens for the case where only one or two features are used. Obviously, such case corresponds to the one of over-elimination of features. In practice, early stopping of backward feature selection would have been triggered by the substantial increase of average test error rate.

Table 5.6 shows the average CPU time over 30 realizations of the real-world data sets needed by the five feature selection methods to produce the ranked list of all features. The times shown exclude the training and testing of SVM for the evaluations of error rates. Two timings are shown: t_{rank}, time needed to yield the full feature ranked lists and t_{cv}, time used in tuning σ in PW and RSDE. An additional timing, t_{qp}, is also included for MI-RSDE and it corresponds to the time needed for the solution of the quadratic optimization problem of (5.2). Note that t_{rank} includes t_{qp} for MI-RSDE.

From Table 5.6, the times needed by mRMR are much smaller than those by the other four methods. This is expected since mRMR uses mutual information involving only two features. The other four methods are somewhat similar in the times needed with

MI-RSDE needed more time than the other three on data sets Abalone, WBCD, Glass and Wine. It is also of interest to note that MI-RSDE spends less time than MI-PW and Kwak on data sets Satimage and Musk. This shows that forward method like Kwak is not always faster than the backward methods.

5.4.3 Discussion

The experiments of the preceding sections suggest that MI-PW is an effective feature selection approach for both artificial and real-world problems. For artificial problems, MI-PW consistently yields better performance than all other methods, and its effectiveness is not affected much when the training set is small. For the real-world problems, MI-PW consistently performs at least as well, if not better than the other methods for all problems.

The better performance of MI-PW over mRMR and Kwak is expected since the latter two methods use the forward feature selection scheme. The better performance justifies the additional computations needed for the estimation of the higher dimensional density functions. It is also interesting to note that HSIC is effective in dealing with data set having interactive features, as shown in the artificial problems, but not as effective as MI-PW and MI-RSDE. It also does not do well on real-world data sets of Abalone, WBCD and Musk. Between the two proposed methods, MI-PW generally performs better than MI-RSDE. This is probably due to inaccuracy of RSDE on complex data sets.

5.5 Summary

A new filter feature selection method based on mutual information functions is proposed in this chapter. Unlike most other filter methods, the proposed method is implemented in a backward selection framework and, hence, is effective in handling data sets with dependency involving multiple features. Numerical experiments of the proposed methods, in comparisons with several benchmark methods, are provided for artificial and real-world data sets. The experiments also show that the proposed method (with PW estimation) has better performances over the other benchmark methods for all the data sets considered. The evaluation of proposed criterion requires estimations of probability density functions using either the PW or the RSDE and is therefore more expensive computationally than some of the benchmark methods. This higher cost is justified in view of its superior performance.

	$\lvert\mathscr{D}_{trn}\rvert$	$\lvert\mathscr{D}_{tst}\rvert$	d	c	n_r
Abalone	1044	3133	8	3	1
WBCD	350	333	9	2	1
Glass	180	34	9	6	1
Wine	120	58	13	3	1
Satimage	2000	4435	36	6	1
Musk	330	146	166	2	1

Table 5.5: Description of real-world data sets for classification.

Dataset	MI-PW		MI-RSDE		Kwak		HSIC	mRMR
	t_{cv}	t_{rank}	t_{cv}	t_{rank} (t_{qp})	t_{cv}	t_{rank}	t_{rank}	t_{rank}
Abalone	0.58	8.19	20.90	9.70 (6.19)	0.58	7.07	13.89	0.15
WBCD	0.10	0.93	4.01	1.02 (0.67)	0.10	0.84	1.21	0.04
Glass	0.04	0.21	1.97	0.34 (0.25)	0.04	0.24	0.20	0.03
Wine	0.03	0.19	1.06	0.50 (0.40)	0.03	0.21	0.17	0.04
Satimage	1.41	813.30	42.20	716.25 (77.36)	1.41	773.24	1377.75	1.85
Musk	0.19	344.69	2.98	223.44 (8.49)	0.19	289.68	366.84	26.60

Table 5.6: Average time (sec) of yielding feature ranking lists by all methods over 30 realizations of real-world data sets.

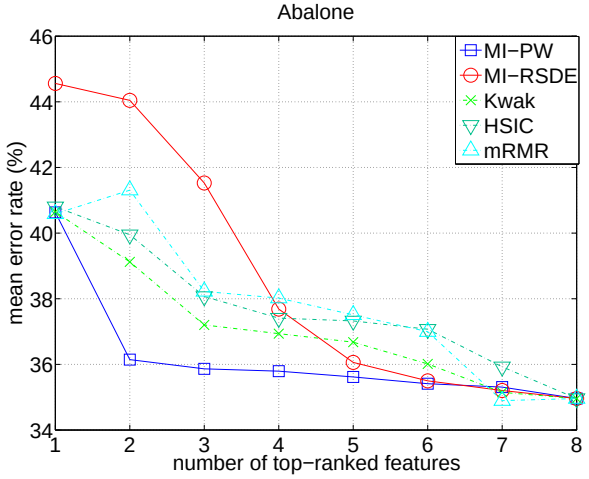


Figure 5.4: Test error rates on Abalone data set

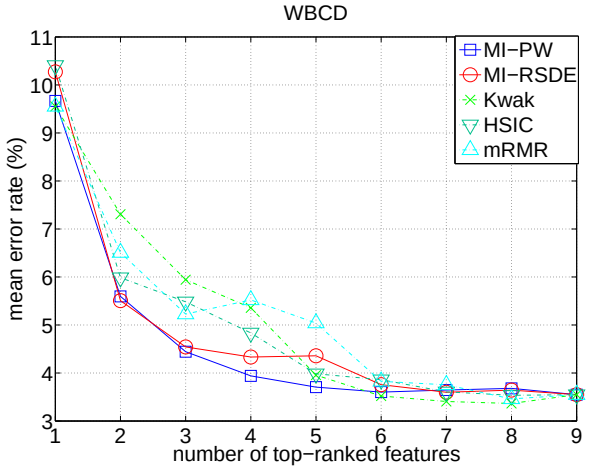


Figure 5.5: Test error rates on WBCD data set

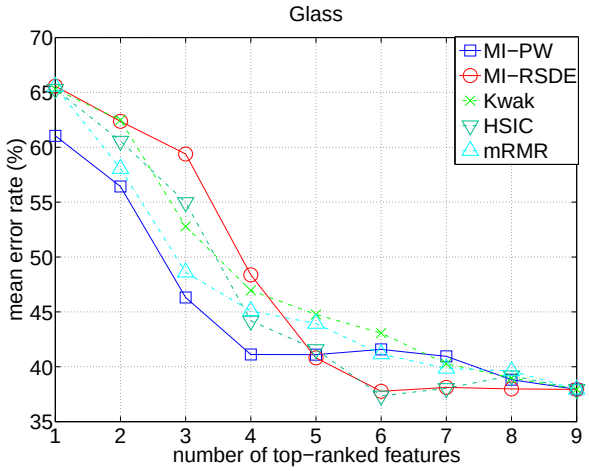


Figure 5.6: Test error rates on Glass data set

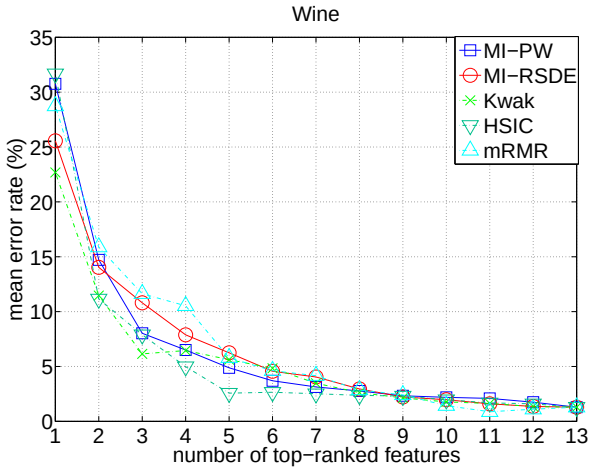


Figure 5.7: Test error rates on Wine data set

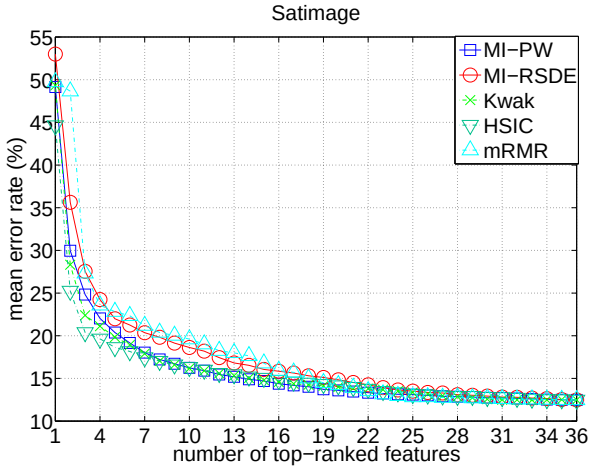


Figure 5.8: Test error rates on Satimage data set

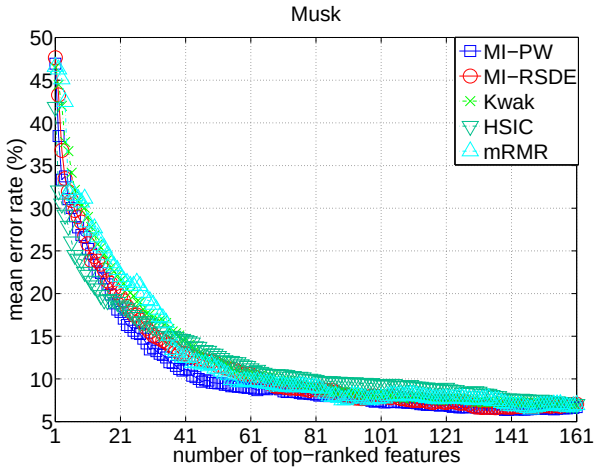


Figure 5.9: Test error rates on Musk data set

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	40.63	44.56	**0.00**+	40.65	0.91	40.80	0.37	40.59	0.74
2	36.14	44.04	**0.00**+	39.12	**0.00**+	39.95	**0.00**+	41.31	**0.00**+
3	35.86	41.53	**0.00**+	37.20	**0.00**+	38.07	**0.00**+	38.22	**0.00**+
4	35.79	37.68	**0.00**+	36.94	**0.00**+	37.41	**0.00**+	38.02	**0.00**+
5	35.62	36.06	0.08	36.68	**0.00**+	37.32	**0.00**+	37.51	**0.00**+
6	35.41	35.50	0.62	36.01	**0.01**+	37.07	**0.00**+	37.00	**0.00**+
7	35.31	35.20	0.46	35.15	0.40	35.93	**0.00**+	34.90	**0.01**+
8	34.96	34.96	1.00	34.96	0.99	34.96	1.00	34.96	0.99

Table 5.7: *t*-test on Abalone data set.

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	9.67	10.27	0.14	9.55	0.75	10.41	0.06	9.55	0.75
2	5.60	5.51	0.80	7.31	**0.00**+	5.99	0.41	6.51	**0.01**+
3	4.44	4.54	0.76	5.94	**0.00**+	5.48	**0.00**+	5.23	**0.01**+
4	3.94	4.33	0.19	5.36	**0.00**+	4.83	**0.01**+	5.52	**0.00**+
5	3.71	4.36	**0.01**+	3.96	0.37	3.98	0.27	5.04	**0.00**+
6	3.60	3.76	0.54	3.52	0.73	3.86	0.26	3.83	0.30
7	3.64	3.60	0.84	3.41	0.29	3.61	0.86	3.75	0.63
8	3.68	3.64	0.88	3.37	0.16	3.54	0.54	3.46	0.33
9	3.55	3.55	1.00	3.55	1.00	3.55	1.00	3.55	1.00

Table 5.8: *t*-test on WBCD data set.

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	61.04	65.57	0.05	65.29	0.06	65.30	0.06	65.53	0.05
2	56.43	62.37	**0.02+**	62.48	**0.01+**	60.57	0.09	58.06	**0.50+**
3	46.31	59.39	**0.00+**	52.79	**0.02+**	54.98	**0.00+**	48.60	0.39
4	41.12	48.37	**0.01+**	46.95	**0.02+**	44.18	0.27	45.09	0.11
5	41.10	40.81	0.92	44.77	0.15	41.58	0.86	43.94	0.29
6	41.58	37.76	0.08	43.07	0.53	37.35	0.09	41.15	0.86
7	40.95	38.12	0.25	40.27	0.79	38.03	0.25	39.85	0.66
8	38.82	37.98	0.72	38.99	0.94	39.20	0.88	39.61	0.76
9	37.94	37.94	1.00	37.94	1.00	37.94	1.00	37.94	1.00

Table 5.9: *t*-test on Glass data set.

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	30.76	25.56	0.08	22.68	**0.00-**	31.67	0.77	28.74	0.51
2	14.72	14.04	0.73	11.46	**0.04-**	11.15	**0.03-**	15.88	0.59
3	8.03	10.80	0.09	6.15	0.07	7.90	0.87	11.62	**0.01+**
4	6.50	7.89	0.37	6.45	0.96	4.98	0.11	10.53	**0.00+**
5	4.88	6.24	0.33	5.62	0.38	2.57	0.05	5.78	0.29
6	3.70	4.57	0.49	4.83	0.10	2.65	0.08	4.62	0.14
7	3.14	4.04	0.34	3.47	0.49	2.52	0.14	4.16	**0.03+**
8	2.77	2.95	0.76	2.62	0.75	2.37	0.30	2.86	0.83
9	2.33	2.22	0.77	2.14	0.65	2.30	0.95	2.41	0.85
10	2.19	1.97	0.64	1.77	0.30	1.98	0.63	1.43	0.05
11	2.09	1.60	0.27	1.63	0.25	1.68	0.33	0.87	0.05
12	1.74	1.36	0.34	1.35	0.30	1.60	0.72	1.11	0.10
13	1.29	1.29	1.00	1.29	1.00	1.29	1.00	1.29	1.00

Table 5.10: *t*-test on Wine data set.

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	49.15	53.00	**0.00+**	49.33	0.81	44.62	**0.00-**	49.73	0.42
4	22.03	24.24	**0.03+**	21.12	0.05	19.64	0.09	23.57	**0.00+**
7	18.02	20.36	**0.00+**	17.90	0.70	17.45	0.06	21.19	**0.00+**
10	16.24	18.63	**0.00+**	16.17	0.82	16.31	0.81	19.61	**0.00+**
13	15.19	16.81	**0.00+**	15.32	0.62	15.46	0.31	18.00	**0.00+**
16	14.38	15.84	**0.00+**	14.57	0.43	14.78	0.10	15.92	**0.00+**
19	13.73	15.11	**0.00+**	14.14	**0.02+**	14.32	**0.00+**	14.49	**0.00+**
22	13.33	14.25	**0.00+**	13.79	**0.01+**	13.76	**0.01+**	13.53	0.19
25	13.10	13.51	**0.01+**	13.36	0.11	13.14	0.78	12.82	0.06
28	12.85	13.09	0.12	12.85	0.99	12.91	0.69	12.78	0.65
31	12.71	12.81	0.48	12.50	0.15	12.77	0.67	12.61	0.47
34	12.53	12.63	0.48	12.49	0.80	12.55	0.91	12.53	0.98
36	12.48	12.48	1.00	12.48	1.00	12.48	1.00	12.48	1.00

Table 5.11: *t*-test on Satimage data set.

No.	MI-PW	MI-RSDE		Kwak		HSIC		mRMR	
	mean value	mean value	p-value	mean value	p-value	mean value	p-value	mean value	p-value
1	46.93	47.63	0.63	46.46	0.77	41.85	**0.00-**	46.58	0.83
11	25.23	25.84	0.61	28.98	**0.00+**	22.17	0.11	28.37	**0.00+**
21	17.85	19.65	**0.03+**	21.53	**0.00+**	18.58	0.35	21.97	**0.00+**
31	13.64	15.10	**0.04+**	17.16	**0.00+**	16.11	**0.00+**	18.71	**0.00+**
41	11.09	12.96	**0.01+**	14.69	**0.00+**	14.51	**0.00+**	12.48	**0.04+**
51	9.47	11.48	**0.01+**	11.94	**0.00+**	12.66	**0.00+**	11.16	**0.02+**
61	8.97	10.52	**0.03+**	10.55	**0.03+**	11.22	**0.00+**	9.65	0.31
71	8.54	9.04	0.40	9.79	**0.05+**	10.47	**0.01+**	9.18	0.26
81	8.16	8.73	0.38	8.86	0.21	9.86	**0.01+**	8.97	0.15
91	7.82	8.09	0.65	8.25	0.46	9.54	**0.01+**	7.64	0.76
101	7.44	7.89	0.46	8.09	0.24	9.26	**0.00+**	7.78	0.54
111	7.20	7.46	0.70	7.96	0.20	9.10	**0.00+**	8.13	0.12
121	6.97	7.23	0.71	7.56	0.30	8.94	**0.00+**	7.74	0.20
131	6.57	6.69	0.85	7.31	0.22	8.48	**0.00+**	7.55	0.09
141	6.51	6.68	0.78	7.19	0.23	7.67	0.06	6.85	0.58
151	6.55	6.85	0.61	7.20	0.25	7.52	0.11	6.87	0.60
161	6.70	7.04	0.59	6.87	0.79	7.04	0.59	6.95	0.69
166	7.00	7.00	1.00	7.00	1.00	7.00	1.00	7.00	1.00

Table 5.12: *t*-test on Musk data set.

Chapter 6

Determination of Global Minimum of Some Common Validation Function in Support Vector Machine

Tuning of the regularization parameter, C, is a well-known process in the implementation of a Support Vector Machine classifier. Such a tuning process uses an appropriate validation function whose value, evaluated over a validation set, is to be optimized for the determination of the optimal C. Unfortunately, the validation functions are not smooth functions of C. This chapter presents a method for obtaining the global optimal solution of these non-smooth validation functions. The method is guaranteed to find the global optimum and relies on the regularization solution path of SVM over a range of C values. When the solution path is available, the computation needed is minimal.

The rest of this chapter is arranged as follows. Section 6.1 provides the formulas of the regularization solution path of SVM over C. Section 6.2 shows the main algorithm for determining the global optimum of the validation function. Section 6.3 provides results of numerical experiment of the proposed algorithm and a comparison with several standard approaches. Summary is given in section 6.4.

6.1 Preliminary

As reviewed in 2.1.1 of Chapter 2, the standard two-class SVM primal problem (SVM-PP) is

$$\begin{aligned} \min_{w,b,\zeta} \ & \frac{1}{2} w'w + C \sum_{i \in \mathscr{I}_{\mathscr{D}}} \zeta_i \\ y_i(w'\phi(x_i) + b) & \geq 1 - \zeta_i, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \\ \zeta_i & \geq 0, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}}, \end{aligned}$$

its Dual problem (SVM-DP) is

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \ & \frac{1}{2} \sum_{i \in \mathscr{I}_{\mathscr{D}}} \sum_{j \in \mathscr{I}_{\mathscr{D}}} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i \in \mathscr{I}_{\mathscr{D}}} \alpha_i \\ & 0 \leq \alpha_i \leq C, \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \\ & \sum_i \alpha_i y_i = 0, \end{aligned}$$

and the output function of SVM is

$$f(x) = \sum_{i \in \mathscr{I}_{\mathscr{D}}} \alpha_i y_i K(x_i, x) + b.$$

It can be seen from the above formulas that C is a parameter in SVM-PP, the solution of SVM-DP in the form of $\{\alpha_i : i \in \mathscr{I}_{\mathscr{D}}\}$ and b are all functions of C. It is possible to numerically determine these solutions for the entire range of C, resulting in a *regularization solution path* of SVM. Works in this direction are given by Hastie et al. [34] and Ong et al. [58]. Hastie et al. [34] provide the framework for the approach following techniques from parametric programming while Ong et al. [58] use a different formulation to improve on the reliability of the algorithm. Among others, Ong et al.'s approach takes into consideration numerical problems that can arise in a data set having nominal features, duplicate points, and/or linearly dependent points in the kernel space. Detailed information of the approach can be found in [34] and [58].

The rest of this section provides a summary of Ong et al.'s approach [58] whose results will be needed in the sequel. To facilitate discussion, the notations used in [34] and [58] are adopted:

$$\lambda := C^{-1}, \quad \alpha_0(\lambda) := b(\lambda), \tag{6.1}$$

$$\hat{\alpha}_i(\lambda) := \lambda \alpha_i(\lambda), \ \ \forall\, i \in \mathscr{I}_{\mathscr{D}} \cup \{0\} \tag{6.2}$$

where the dependence of $\hat{\alpha}_i$ and α_i on λ (equivalently, C) are shown explicitly. The solution of SVM dual problem (DP) reviewed in 2.1.1 of Chapter 2 at any specific value

of λ consists of the optimal $\hat{\alpha}_i(\lambda), i \in \mathscr{I}_{\mathscr{D}} \cup \{0\}$. Because of the constraint $0 \leq \alpha_i \leq C, \forall\, i \in \mathscr{I}_{\mathscr{D}}$ and (6.2), $\hat{\alpha}_i(\lambda)$ takes value between 0 and 1 for all $i \in \mathscr{I}_{\mathscr{D}}$. Hence, it is convenient to introduce the following mutually exclusive sets

$$\mathscr{R}(\lambda) := \{i \in \mathscr{I}_{\mathscr{D}} : \hat{\alpha}_i(\lambda) = 0\},\ \ \mathscr{L}(\lambda) := \{i \in \mathscr{I}_{\mathscr{D}} : \hat{\alpha}_i(\lambda) = 1\}$$

$$\text{and } \mathscr{E}(\lambda) := \{i \in \mathscr{I}_{\mathscr{D}} : 0 < \hat{\alpha}_i(\lambda) < 1\}$$

with the property that $\mathscr{R}(\lambda) \bigcup \mathscr{L}(\lambda) \bigcup \mathscr{E}(\lambda) = \mathscr{I}_{\mathscr{D}}$ at every λ.

The algorithm in [58], known as Improved SVM Path (ISVMP), starts with a user-defined range of λ, $(\overline{\lambda}, \underline{\lambda})$, over which SVM solution path is needed. Typically, $(\overline{\lambda}, \underline{\lambda})$ is a large interval that covers the range of interest. The output of ISVMP consists of a set of critical values of λ in

$$\Lambda := \{\lambda^0, \cdots, \lambda^{\ell_{\max}}\} \tag{6.3}$$

with $\lambda^0 := \overline{\lambda}$, $\lambda^{\ell_{\max}} = \underline{\lambda}$, $\lambda^\ell > \lambda^{\ell+1}$ and the corresponding

$$\{\hat{\alpha}_i(\lambda^\ell) : i \in \mathscr{I}_{\mathscr{D}} \cup \{0\}\} \text{ for every } \lambda^\ell \in \Lambda. \tag{6.4}$$

Each critical λ value corresponds to a qualitative changes in the SVM solutions: elements in $\mathscr{R}(\lambda), \mathscr{L}(\lambda)$ or $\mathscr{E}(\lambda)$ changes when λ crosses over λ^ℓ. More exactly, each $\lambda^\ell \in \Lambda$ corresponds to the occurrence of one of the following events:

- an index $i \in \mathscr{E}(\lambda^{\ell+})$ moves to $\mathscr{L}(\lambda^\ell)$ or $\mathscr{R}(\lambda^\ell)$,

- an index $i \in \mathscr{L}(\lambda^{\ell+})$ moves to $\mathscr{E}(\lambda^{\ell})$,
- an index $i \in \mathscr{R}(\lambda^{\ell+})$ moves to $\mathscr{E}(\lambda^{\ell})$,

where $\lambda^{\ell+}$ refers to value of λ that is slightly larger than λ^{ℓ}.

In Ong et al.'s method, determination of next event $\ell+1$, given the result at event ℓ, is posed in the following linear programming (LP) problem:

$$\min_{\delta_\lambda} \quad \delta_\lambda \tag{6.5}$$

$$\text{s.t. } 0 \leq \ \hat{\alpha}_i^{\ell} + d_p^i \delta_\lambda \leq 1 \ \forall \ i \in \mathscr{E} \tag{6.6}$$

$$(d_p' \hat{k}_i - 1)\delta_\lambda - \lambda^{\ell} \xi_i^{\ell} \geq 0 \ \forall \ i \in \mathscr{R} \tag{6.7}$$

$$(d_p' \hat{k}_i - 1)\delta_\lambda - \lambda^{\ell} \xi_i^{\ell} \leq 0 \ \forall \ i \in \mathscr{L} \tag{6.8}$$

$$\delta_\lambda \geq -\lambda^{\ell} \tag{6.9}$$

where

$$\delta_\lambda = \lambda - \lambda^{\ell} \tag{6.10}$$

$$k_{uv} = K(x_u, x_v) y_u y_v, \ \ \forall u, v \in \mathscr{I}_{\mathscr{D}} \tag{6.11}$$

$$\xi_i(\hat{\alpha}, \hat{\alpha}_0, \lambda) = 1 - \frac{\sum_{j \in \mathscr{I}_{\mathscr{D}}} \hat{\alpha}_j k_{ji} - y_i \hat{\alpha}_0}{\lambda}, \quad \forall i \in \mathscr{I}_{\mathscr{D}} \tag{6.12}$$

$$\hat{k}_i = [-y_i,\ k_{1i}, \cdots,\ k_{|\mathscr{E}|i}]' \tag{6.13}$$

and

$$d_p = A^{-1}\mathbf{1} = \begin{pmatrix} 0 & -y_1 & \cdots & -y_{|\mathscr{E}|} \\ -y_1 & k_{11} & \cdots & k_{1|\mathscr{E}|} \\ \vdots & \vdots & \vdots & \vdots \\ -y_{|\mathscr{E}|} & k_{|\mathscr{E}|1} & \cdots & k_{|\mathscr{E}||\mathscr{E}|} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{6.14}$$

For notional convenience, $\{1, \cdots, |\mathscr{E}|\}$ in (6.13) and (6.14) refer to all indices in $\mathscr{E}$. The constrains (6.6) to (6.8) are imposed to ensure that all SVM solutions from event ℓ to event $\ell+1$ satisfy KKT conditions shown in Section 2.1.1, while constrain (6.9) is imposed to ensure that only $\lambda \geq 0$ is considered. Supposing δ^*_λ is the minimizer of LP, λ at the next event is defined by $\lambda^{\ell+1} = \delta^*_\lambda + \lambda^\ell$.

Note that the formulas of LP model (6.5) to (6.9) and d_p solution (6.14) are for the case that the square matrix A in (6.14) is invertible. For the case that A is not invertible, some modifications on LP model and d_p solution are needed. The details of them can be found in [58].

The sets given by (6.3) and (6.4) fully characterize the solution path of SVM. For λ such that $\lambda^{\ell+1} < \lambda \leq \lambda^\ell$, $\hat{\alpha}_i(\lambda)$ for any $i \in \mathscr{I}_\mathscr{D} \cup \{0\}$ can be found by interpolation using

$$\hat{\alpha}_i(\lambda) := \frac{\lambda^{\ell+1} - \lambda}{\lambda^{\ell+1} - \lambda^\ell}\hat{\alpha}_i(\lambda^\ell) + \frac{\lambda - \lambda^\ell}{\lambda^{\ell+1} - \lambda^\ell}\hat{\alpha}_i(\lambda^{\ell+1}). \tag{6.15}$$

The above solution path of SVM over a range of C is fully utilized to select the global optimal regularization parameter C in the subsequent sections.

6.2 Finding the Global Optimal Solution

Consider a given validation set denoted by $V := \{(x_i, y_i) : i \in \mathscr{I}_V\}$. The output function of $f(\cdot)$ at a specific value of λ can be expressed as

$$f(x,\lambda) = \frac{1}{\lambda}\Big(\sum_{i\in\mathscr{I}_{\mathscr{D}}} \hat{\alpha}_i(\lambda) y_i z_i \cdot z_j + \hat{\alpha}_0(\lambda)\Big). \tag{6.16}$$

where $z_i := \phi(x_i)\ \forall i \in \mathscr{I}_{\mathscr{D}} \cup \mathscr{I}_V$. The tuning process involves finding the optimal λ value of a validation function on V which requires frequent evaluation of $f(x_j,\lambda)$ for j in $\mathscr{I}_V$. For convenience, define

$$h_j(\lambda) := \lambda f(x_j,\lambda) = \sum_{i\in\mathscr{I}_{\mathscr{D}}\cup\{0\}} \hat{\alpha}_i(\lambda) g_{ij} \tag{6.17}$$

$$= \sum_{i\in\mathscr{L}(\lambda)} g_{ij} + \sum_{i\in\mathscr{E}(\lambda)\cup\{0\}} \hat{\alpha}_i(\lambda) g_{ij} \tag{6.18}$$

where $g_{ij} := y_i z_i \cdot z_j$ for any $(i,j) \in \mathscr{I}_{\mathscr{D}} \times \mathscr{I}_V$ and $g_{0j} = 1$ for all $j \in \mathscr{I}_V$. Equation (6.18) follows from (6.17) because $\hat{\alpha}_i(\lambda) = 0$ and 1 for $i \in \mathscr{R}(\lambda)$ and $\mathscr{L}(\lambda)$ respectively. Since $h_j(\lambda)$ and $\lambda f(x_j,\lambda)$ have the same sign, the predicted output class of $x_j \in V$ is

$$\tilde{y}_j(\lambda) := \text{sign}(h_j(\lambda)) = \begin{cases} +1, & \text{if } h_j(\lambda) \geq 0, \\ -1, & \text{if } h_j(\lambda) < 0. \end{cases}$$

The proposed approach is applicable to the various validation functions including error rate, weighted error rate, precision (percentage of positive predictions that are correct), recall (percentage of positive validation examples that are correctly predicted), F measure (harmonic mean of precision and recall) and area under ROC curve. However, the steps involved are best illustrated using one choice of validation function. Extensions of the approach to other validation functions and cross-validation set are discussed in *Remarks* 6.2.1 and 6.2.2. Our choice corresponds to probably the most common validation function, namely the error rate function, given by

$$E(\lambda) = \frac{1}{2|V|} \sum_{j \in \mathscr{I}_V} |y_j - \tilde{y}_j(\lambda)|, \tag{6.19}$$

which measures the percentage of incorrect predictions.

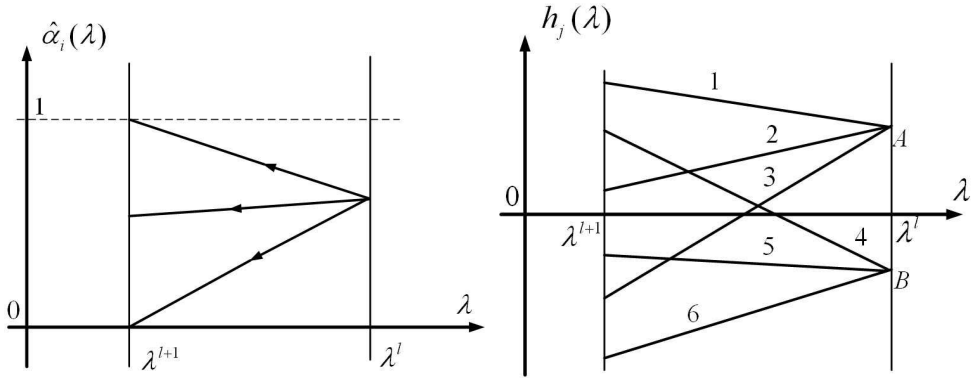
The proposed approach relies on the following facts:

(a) $E(\lambda)$ is a piecewise-constant function of λ and changes value only when at least one $\tilde{y}_j(\lambda)$ changes value.

(b) $\tilde{y}_j(\lambda)$ changes value only when $h_j(\lambda)$ crosses the zero value, either from positive to negative or vice versa.

(c) $h_j(\lambda)$ depends affinely on λ for $\lambda^\ell \geq \lambda > \lambda^{\ell+1}$, following (6.15) and (6.18).

From (a) and (b), an important aspect of finding the global optimum of $E(\lambda)$ is to find the value of λ at which $h_j(\lambda)$ crosses the zero value. For this purpose, consider the values of $\hat{\alpha}_i(\lambda)$ and $h_j(\lambda)$ between λ^ℓ and $\lambda^{\ell+1}$. Figures 6.1(a) and 6.1(b) show the



(a) (b)

Figure 6.1: (a) Typical values of $\hat{\alpha}_i(\lambda), i \in \mathscr{E}(\lambda^\ell)$ for $\lambda^{\ell+1} < \lambda \leq \lambda^\ell$. (b) Typical values of $h_j(\lambda)$ for $\lambda^{\ell+1} < \lambda \leq \lambda^\ell$. Points A and B refer to two possible values of $h_j(\lambda^\ell)$, positive and negative.

possible plots of $\hat{\alpha}_i(\lambda)$ and $h_j(\lambda)$ as a function of λ in this interval respectively. For a change in the value of $E(\lambda)$, it follows from (b) that at least one $h_j(\lambda)$ among $j \in \mathscr{I}_V$ must have a zero-crossover. This also means that $h_j(\lambda)$ is of Type 3 or 4 in Figure 6.1(b). Hence, a point j causes a change in $E(\lambda)$ if and only if $h_j(\lambda^\ell)$ and $h_j(\lambda^{\ell+1})$ have different sign. Let the collection of such points be

$$\mathscr{I}_S^\ell = \{j: \ \ h_j(\lambda^\ell) \cdot h_j(\lambda^{\ell+1}) < 0, \ \ j \in \mathscr{I}_V\}. \tag{6.20}$$

From (c), a convenient representation of $h_j(\lambda)$ is

$$h_j(\lambda) = \frac{\lambda^{\ell+1} - \lambda}{\lambda^{\ell+1} - \lambda^\ell} h_j^\ell + \frac{\lambda - \lambda^\ell}{\lambda^{\ell+1} - \lambda^\ell} h_j^{\ell+1}, \ \lambda^{\ell+1} < \lambda \leq \lambda^\ell \tag{6.21}$$

where $h_j^\ell := h_j(\lambda^\ell)$. Using this expression, the zero-crossover of $h_j(\lambda)$ for $\lambda^{\ell+1} < \lambda \leq$

λ^{ℓ} happens at

$$\lambda_j^{\ell*} = \frac{\lambda^{\ell+1}h_j^{\ell} - \lambda^{\ell}h_j^{\ell+1}}{h_j^{\ell} - h_j^{\ell+1}}, \ \forall j \in \mathscr{I}_S^{\ell}. \tag{6.22}$$

Let these indices of $\lambda_j^{\ell*}$ be collected into an ordered set

$$\mathscr{I}_{\lambda}^{\ell} = \{i_1, i_2, \cdots, i_{|\mathscr{I}_S^{\ell}|}\} \tag{6.23}$$

such that $\lambda_{i_1}^{\ell*} \geq \lambda_{i_2}^{\ell*} \geq \cdots \geq \lambda_{i_{|\mathscr{I}_S^{\ell}|}}^{\ell*}$.

With (6.22), it is possible to update $E(\lambda)$ when λ crosses $\lambda_j^{\ell*}$. To see this, suppose the value of $E(\lambda^{\ell})$ is known, it follows from (6.19) that

$$E(\lambda) = \frac{1}{2|V|} \sum_{j \in \mathscr{I}_S^{\ell}} |y_j - \tilde{y}_j(\lambda)| + \text{constant}, \text{for } \lambda^{\ell} \geq \lambda > \lambda^{\ell+1}.$$

Let $\lambda_{i_m}^{\ell*+}$, $i_m \in \mathscr{I}_{\lambda}^{\ell}$, be the value of λ slightly larger than $\lambda_j^{\ell*}$. Then

$$E(\lambda_{i_m}^{\ell*}) = E(\lambda_{i_m}^{\ell*+}) + \frac{1}{2|V|}\{|y_{i_m} - \tilde{y}_{i_m}(\lambda_{i_m}^{\ell*})| - |y_{i_m} - \tilde{y}_{i_m}(\lambda_{i_m}^{\ell*+})|\}.$$

Since $E(\lambda)$ is a piecewise constant function, $E(\lambda_{i_m}^{\ell*+})$ is a constant for all λ s.t., $\lambda_{i_{m-1}}^{\ell*} \geq \lambda \geq \lambda_{i_m}^{\ell*+}$ with $i_m,\ i_{m-1} \in \mathscr{I}_{\lambda}^{\ell}$. Hence the above can also be modified as

$$E(\lambda_{i_m}^{\ell*}) = \begin{cases} E(\lambda_{i_{m-1}}^{\ell*}) + \frac{1}{|V|}, & \text{if } y_{i_m} = \tilde{y}_{i_m}(\lambda_{i_{m-1}}^{\ell*}) \\ E(\lambda_{i_{m-1}}^{\ell*}) - \frac{1}{|V|}, & \text{otherwise} \end{cases} \tag{6.24}$$

for $m = 1, \cdots, |\mathscr{I}_S^{\ell}|$ and $\lambda_{i_0} = \lambda^{\ell}$ when $m = 0$. Using (6.22), (6.23), (6.24), (6.3) and

(6.4), $E(\lambda)$ can be computed for all $\underline{\lambda} \leq \lambda \leq \overline{\lambda}$.

It is now possible to state the Pseudo code for the overall algorithm. The algorithm assumes that the solution path in the form of (6.3) and (6.4) are available for $\overline{\lambda}$ to $\underline{\lambda}$. The output is the optimal λ, λ^* and the corresponding $E^* := E(\lambda^*)$.

Table 6.1: Pseudo Code

Input: $\overline{\lambda}$, $\underline{\lambda}$, $\ell_{\max}$, Λ, $\mathscr{D}$, V and $\{\hat{\alpha}_i(\lambda) : i \in \mathscr{I}_{\mathscr{D}} \cup \{0\}, \lambda \in \Lambda\}$
Output: E^* and λ^*
1. Initialization:
  Let $g_{0j} = 1$, $\forall j \in \mathscr{I}_V$ and $\lambda^0 = \overline{\lambda}$.
  Compute:
    $g_{ij} = y_i z_i \cdot z_j$, $\forall i \in \mathscr{I}_{\mathscr{D}}, j \in \mathscr{I}_V$,
    h_j^ℓ using (6.17) $\forall \ell = 0, 1, \cdots, \ell_{\max}$ and $\forall j \in \mathscr{I}_V$,
    $E(\lambda^0)$ from (6.19).
  Let $E^* = E(\lambda^0)$, $\lambda^* = \lambda^0$ and $\ell = 0$
2. Main loop:
  While $\ell < \ell_{\max}$,
    a. Read in $\lambda^{\ell+1}$ and $\{h_j^{\ell+1} : j \in \mathscr{I}_V\}$.
    b. Compute:
      $h_j^{\ell+1} \cdot h_j^\ell$, $\forall j \in \mathscr{I}_V$ and form $\mathscr{I}_S^\ell$ using (6.20).
      $\lambda_j^{\ell *}$ using (6.22) $\forall j \in \mathscr{I}_S^\ell$ and form $\mathscr{I}_\lambda^\ell$ of (6.23).
    c. For each $i_m \in \mathscr{I}_\lambda^\ell$ starting from i_1,
      Compute $E(\lambda_{i_m}^{\ell *})$ using (6.24),
      If $E(\lambda_{i_m}^{\ell *}) < E^*$,
        then let $E^* = E(\lambda_{i_m}^{\ell *})$ and $\lambda^* = \lambda_{i_m}^{\ell *}$
    d. Let $\ell = \ell + 1$
  end

Remark 6.2.1. *The above exposition is for the validation function given by (6.19). This validation function can also be expressed as* $E(\lambda) = \frac{1}{2|V|} \sum_{j \in \mathscr{I}_V} \max\{0, 1 - y_j \tilde{y}_j(\lambda)\}$ *which is related to the hinged loss function in SVM. The above development is also applicable, with minor modifications, when E is given by*

- *the Weighted Error rate with* $E(\lambda) = \frac{1}{2(n_+ + \eta n_-)} [\sum_{j \in \mathscr{I}_V^+} |y_j - \tilde{y}_j(\lambda)| + \sum_{j \in \mathscr{I}_V^-} \eta |y_j - \tilde{y}_j(\lambda)|]$ *for some* $\eta > 0$ *where* $n_+ (n_-)$ *is the total number of validation samples*

with $y=+1$ $(y=-1)$ respectively and $\mathscr{I}_V^+$ $(\mathscr{I}_V^-)$ is the subset of indices in $\mathscr{I}_V$ with $y=+1(y=-1)$ respectively. The Weighted Error rate becomes the Balanced Error rate when $\eta=\frac{n_+}{n_-}$.

- *the Precision (percentage of positive predictions that are correct) with $E(\lambda)=1-\frac{1}{2N_+(\lambda)}\sum_{j\in\mathscr{I}_V^-}|y_j-\tilde{y}_j(\lambda)|$ where $N_+(\lambda)$ is the total number of j with $\tilde{y}_j(\lambda)=1$.*
- *Recall (percentage of positive validation examples that are correctly predicted) with $E(\lambda)=\frac{1}{2n_+}\sum_{j\in\mathscr{I}_V^+}(2-|y_j-\tilde{y}_j(\lambda)|)$.*
- *F measure (harmonic mean of precision and recall) with $E(\lambda)=\frac{1}{n_++N_+(\lambda)}\sum_{j\in\mathscr{I}_V^+}(2-|y_j-\tilde{y}_j(\lambda)|)$.*

It is quite easy to see that these functions change their values whenever there is a zero-crossover of h_j.

Remark 6.2.2. *In the event that V is one fold of a n-fold data used in a cross validation process, a few changes are needed. More exactly, there is a regularization solution path for each holdout fold, obtained using the $(n-1)$ remaining folds as $\mathscr{D}$. The procedures to compute $\mathscr{I}_S^\ell$ and $\mathscr{I}_\lambda^\ell$ for each holdout fold are exactly the same as that given by (6.20) and (6.23). The only additional requirement is to evaluate E on a denser grid of λ in order to find its global optimal solution. Let $\bar{\Lambda}_k:=\{\lambda_{i_m}^{\ell*}: i_m\in\mathscr{I}_\lambda^\ell,\ell=0,1,\cdots,\ell_{\max}-1$ for the k^{th} holdout fold $\}$ and $\bar{\Lambda}:=\cup_{k=1,\cdots,n}\bar{\Lambda}_k$ such that it contains the λ values of all zero-crossovers of all holdout folds. To find the global optimum, the cross-validation function, $E(\lambda)=E^1(\lambda)+\cdots+E^n(\lambda)$, has to be evaluated for all $\lambda\in\bar{\Lambda}$. The evaluation of $E^k(\lambda)$ for $\lambda\in\bar{\Lambda}_k$ is given by (6.24). To evaluate $E^k(\lambda)$ over $\bar{\Lambda}$ is trivial since $E^k(\lambda)$*

is a piecewise constant function and changes value only at $\lambda \in \bar{\Lambda}_k$. Of course, the final SVM model is one that is obtained using $\mathscr{D}$ as the training data and with λ obtained by the above procedure.

6.3 Numerical Experiment and Discussion

For easy referencing, the proposed method is termed GO, Global Optimal approach. This section compares GO with two standard tuning processes: the grid search method (GRID) and the gradient based method (GRAD). The GRID method computes $E(\lambda)$ over a grid of λ values and chooses the minimum among them. The GRAD method works only on smooth validation functions and requires expression of the gradient of the smooth validation function with respect to λ. For this reason, approximation of $E(\lambda)$ by a smooth function proposed by Keerthi et al. [45] is used. Details of this approximation are given in the Appendix C. Following [45], the numerical routine used in GRAD is LBFGS [9][1].

In all experiments, the optimal λ is chosen from the range $[2^{-8}, 2^{9}]$. Three levels of resolution are used in GRID: 2^{-1}, $2^{-0.1}$ and $2^{-0.01}$ and are termed GRID-1, GRID-0.1 and GRID-0.01, respectively. Like most nonlinear programming methods, LBFGS solution depends on the initial choice of λ. Our experiments use five different initial values, $\{100, 10, 1, 0.1, 0.01\}$, for each data set and their results are indicated by GRAD-*m* where *m* is the initial value. In addition, the smooth validation function for GRAD is

[1] Downloadable from http://www.cs.toronto.edu/∼liam/software.shtml.

$\widetilde{E}$, given in Appendix C.

For consistency in comparison, the time needed to compute the SVM solutions and the computations of h_j^ℓ, $\forall j \in \mathscr{I}_V$ and $\forall \ell \in \bar{\Lambda}$ is removed from all three methods. This means that the complete regularization solution path from $\bar{\lambda}$ to $\underline{\lambda}$ is run once and its solution with h_j^ℓ $\forall j \in \mathscr{I}_V$, $\forall \ell \in \bar{\Lambda}$ is made available to all three methods. Such an approach eliminates the uncertainties associated with the SVM routines. Note that if this is not done, SVM solution for the GRID method will have to be invoked 18–1800 times while GRAD requires the SVM solutions depending on the number of intermediate λ used by the LBFGS algorithm. Of course, GO uses the entire regularization path while GRID and GRAD need SVM solutions at some selected values of λ. As an approximate guide, timing needed for one SVM regularization path is about the same as that needed for several calls (2-8) to SVM solutions [34, 58] for most data sets.

Numerical experiments are done on Intel Pentium D 3.0G Hz with 1.5G memory under the Linux operating system. The regularization solution path is obtained using ISVMP [58] matlab code (Matlab 2009) available from http://guppy.mpe.nus.edu.sg/∼mpeongcj/ongcj.html. The data sets and their characteristics are given in Table 6.2 and are obtained from [1] and http://www.csie.ntu.edu.tw/ ∼cjlin/libsvmtools/datasets/. For each data set, the experiments are conducted over 10 realizations. The $\ell_{\max}$ for the first realization is indicated in Table 6.2. Each of the 10 realizations is created by random (stratified) sampling of the given set into $\mathscr{D}_{trn}$ and $\mathscr{D}_{tst}$ in the ratio of $|\mathscr{D}_{trn}| : |\mathscr{D}_{tst}| = 3 : 1$. In each method, $\mathscr{D}_{trn}$ is used in a 5-fold cross-validation procedure to determine the optimal C while $\mathscr{D}_{tst}$ is a test set for performance evaluation. For each realization, $\mathscr{D}_{trn}$ is normal-

ized to zero mean and unit standard deviation and its normalization parameters are then used to normalize $\mathscr{D}_{tst}$. All experiments are done using Linear kernel.

Table 6.3 shows the optimal λ^* and the 5-fold cross-validation error, E^*, obtained by each method on the first realization. Note that while $\widetilde{E}$ is the validation function of the GRAD method, the values shown in the table are those of E evaluated at λ^*. Several observations are clear. First, the proposed method obtains the global minimal solution for all 14 data sets. The GRID-i methods do so for 71% to 100% of the data sets while GRAD-i methods do so around 36% to 43% of the data sets. Second, there are data sets where the minimal E^* are obtained at multiple values of λ. For these case, GO always returns the largest value of λ^*. This is not so for the GRAD methods. The larger value of λ^* (or smaller value of C) is advantageous as it yields better generalization performance [81]. Third, there are many cases for which GRAD-*m* returns the initial λ values as the optimal. This is not too surprising since $E(\lambda)$ is a piecewise-constant function with many ranges of λ having gradients that are very close to 0 (termination condition for LBFGS). This situation is clearly depicted in Figure 6.2. The figure also shows that the 5-fold cross-validation error (solid line) is quite different from the smooth 5-fold error function (dashed line) obtained from $\widetilde{E} = \frac{1}{5}\sum_{i=1}^{5}\widetilde{E}_i$. This discrepancy, we believe, is due to the choice of the parameters used in $\widetilde{E}$ (see Appendix C) which is less sensitive to variation of $E(\lambda)$ at small values of λ. While the GRID-0.01 result can also obtain the global optimum of E for the data sets considered, there is no mechanism in it to ensure this performance for other data sets, unlike GO.

For generalization performance, the SVM classifier with λ^* obtained by the various

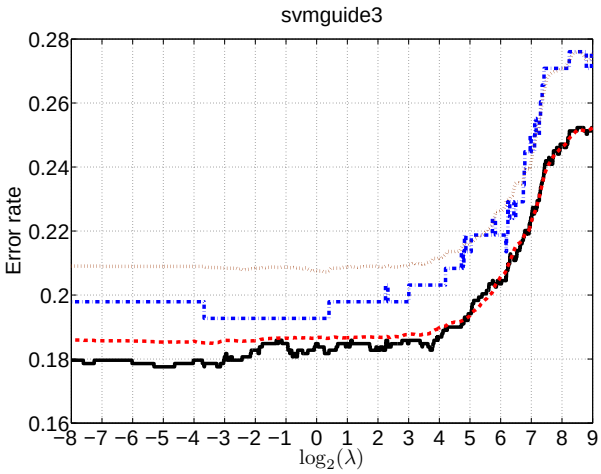


Figure 6.2: Curves of cross-validation error rates (CVER) as functions of λ for data set svmguide3. Solid line - 5-fold CVER; Dashed line - smooth 5-fold CVER; Dashed-dot line - CVER of fold 1; Dot line - smooth CVER of fold 1. The CVER functions for the other folds are omitted to prevent clutter. The optimal λ is 0.114 or $\log_2(0.114) = -3.1329$.

methods are evaluated on $\mathscr{D}_{tst}$. Table 6.4 shows these test error rates $E^\dagger$ of the first realization. It shows that GO yields the lowest test error rate among all methods for all data sets. The GRID-i does so for 86% to 100% of the data sets while GRAD-i averages around 57% to 79% of the data sets. There are some minor variations in the results for the other realizations. Table 6.5 shows the mean and standard derivation values of $E^\dagger$ of all methods over the 10 realizations. Three methods GO, GRID-0.1 and GRAD-1 have the lowest mean test error rate in 8 of the 14 data sets and their performances are better than the others. It is also interesting to note that in data sets heart, monk-1 and hillvalley, GO yields smaller standard derivations than the method with the lowest mean test error rate.

When the SVM solution path is available, the computations needed to compute the λ^* is quite efficient. For each realization of the data sets, the computational time needed by GO to obtain λ^* using 5-fold cross validation process ranges from 3 milliseconds

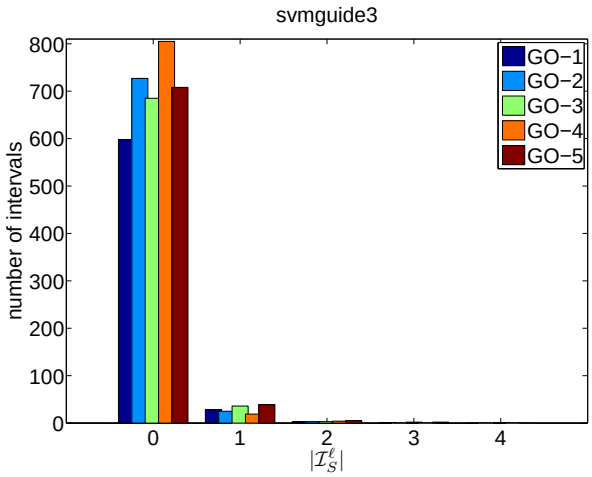


Figure 6.3: The histogram of intervals having various values of $|\mathscr{I}_S^\ell|$ for the 5 folds of svmguide3 in the first realization. The set $|\Lambda_k|$ for $k = 1$ to 5 are $630, 755, 727, 828$ and 754 respectively.

to 8 seconds. These numbers are generally higher than that by GRID-m and GRAD-m. However, since GO is implemented in Matlab while GRID and GRAD are in C, comparison by CPU timing may not be meaningful. Another useful measure is the estimate of the computational complexity of the algorithm with respect to $|\mathscr{I}_V|$ and $\ell_{\max}$. Main computations needed by the algorithm are those associated with (6.20), (6.22) and (6.24). These are proportional to $|I_V|$, $|\mathscr{I}_S^\ell|$ and $\ell_{\max}$. The determination of $|\mathscr{I}_S^\ell|$ of (6.20) for $\ell_{\max}$ events is $O(|I_V| \cdot \ell_{\max})$. The computation of (6.22) and (6.24) depends on the size of $|\mathscr{I}_S^\ell|$. For this purpose, it is useful to know the distribution of $|\mathscr{I}_S^\ell|$ over ℓ. Figure 6.3 shows the histogram (number of intervals) with increasing values of $|\mathscr{I}_S^\ell|$ for the SVMguide3 data set for the 5-fold cross-validation error. As shown, $|\mathscr{I}_S^\ell| = 0$ for more than 90% of all intervals. The histogram shown is typical of other data sets and realizations. Hence, the computations of (6.22) and (6.24) are much smaller than that required for (6.20) which means that the computational complexity is $O(|I_V| \cdot \ell_{\max})$. The dependence of $\ell_{\max}$ on $|\mathscr{D}|$ varies greatly, see [58] for details.

Data set	$\lvert\mathscr{I}_{\mathscr{D}}\rvert$	$\lvert\mathscr{I}_V\rvert$	d	$\ell_{\max}$
colon	47	15	2000	8
leukemia	54	18	7129	1
sonar	138	69	60	286
heart	180	90	13	315
ionosphere	234	117	33	534
wbcd	455	228	9	495
monk 1	370	186	6	596
monk 2	400	201	6	1
monk 3	369	185	6	884
diabetes	512	256	8	407
hillvalley	808	404	100	1021
german	667	333	24	400
svmguide3	856	428	22	861
splice	2382	793	60	3637

Table 6.2: Characteristics of data sets used in the experiments.

6.4 Summary

This chapter describes an approach to obtain the global optimum of the validation function for SVM classifier for the regularization parameter, C. This is possible because the SVM solution path for a range of C can be computed. All existing methods either obtain a local minimum via an approximation of the validation function or a minimum over a set of discrete values of C. The algorithm requires the solution of the SVM solution path. When that is done, the timing needed for the approach is comparable to existing methods and is generally very efficient. In the case when there are multiple C values that attain the global optimum of the validation function, the smallest C value is returned by the approach.

Dataset	GO		GRID-1		GRID-0.1		GRID-0.01		GRAD-100		GRAD-10		GRAD-1		GRAD-0.1		GRAD-0.01	
	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*	λ^*	E^*
colon	512.000	**0.156**	512.000	**0.156**	512.000	**0.156**	512.000	**0.156**	100.000	0.196	10.000	0.196	1.000	0.196	0.100	0.196	0.010	0.196
leukemia	512.000	**0.034**	512.000	**0.034**	512.000	**0.034**	512.000	**0.034**	100.000	**0.034**	10.000	**0.034**	1.000	**0.034**	0.100	**0.034**	0.010	**0.034**
sonar	87.762	**0.000**	64.000	**0.000**	84.449	**0.000**	87.427	**0.000**	26.481	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
heart	83.464	**0.134**	64.000	**0.134**	78.793	**0.134**	83.286	**0.134**	95.183	0.144	9.763	0.144	1.184	0.143	0.100	0.143	0.356	0.143
ionosphere	104.860	**0.000**	64.000	**0.000**	103.970	**0.000**	104.690	**0.000**	79.913	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
wbcd	55.357	**0.000**	32.000	**0.000**	51.984	**0.000**	55.330	**0.000**	8.431	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
monk 1	323.270	**0.283**	1.000	0.288	315.170	0.287	321.800	**0.283**	100.000	0.331	10.000	0.331	1.000	0.288	0.736	0.292	0.607	0.292
monk 2	512.000	**0.348**	512.000	**0.348**	512.000	**0.348**	512.000	**0.348**	100.000	**0.348**	10.000	**0.348**	1.000	**0.348**	0.100	**0.348**	0.010	**0.348**
monk 3	90.586	**0.183**	64.000	0.195	90.510	**0.183**	90.510	**0.183**	101.850	0.207	10.000	0.209	1.000	0.209	0.004	0.209	0.004	0.209
diabete	64.735	**0.217**	64.000	**0.217**	64.000	**0.217**	64.445	**0.217**	48.731	0.219	10.143	0.227	1.000	0.236	0.100	0.234	0.010	0.236
hillvalley	0.004	**0.289**	0.004	**0.289**	0.004	**0.289**	0.004	**0.289**	99.964	0.533	1.226	0.404	0.910	0.396	0.101	0.359	0.011	0.308
german	43.454	**0.231**	1.000	0.232	42.224	0.232	43.411	**0.231**	30.248	0.235	40.438	0.232	2.128	0.233	0.100	0.233	0.010	0.233
svmguide3	0.114	**0.178**	0.031	**0.178**	0.109	**0.178**	0.113	**0.178**	98.481	0.214	10.115	0.186	0.883	0.183	0.102	0.179	0.026	**0.178**
splice	195.480	**0.153**	256.000	0.156	194.010	0.154	195.360	**0.153**	100.000	0.157	10.093	0.160	1.000	0.159	0.100	0.161	0.010	0.162

Table 6.3: Optimal λ value and 5-fold cross-validation error rates for GO, GRID-i and GRAD-i of the first realization. The smallest error rate for each data set is highlighted in bold.

SINGAPORE

Dataset	GO		GRID-1		GRID-0.1		GRID-0.01		GRAD-100		GRAD-10		GRAD-1		GRAD-0.1		GRAD-0.01	
	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$	λ^*	$E^\dagger$
colon	512.000	**0.067**	512.000	**0.067**	512.000	**0.067**	512.000	**0.067**	100.000	**0.067**	10.000	**0.067**	1.000	**0.067**	0.100	**0.067**	0.010	**0.067**
leukemia	512.000	**0.000**	512.000	**0.000**	512.000	**0.000**	512.000	**0.000**	100.000	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
sonar	87.762	**0.000**	64.000	**0.000**	84.449	**0.000**	87.427	**0.000**	26.481	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
heart	83.464	**0.164**	64.000	**0.164**	78.793	**0.164**	83.286	**0.164**	95.183	**0.164**	9.763	0.179	1.184	0.179	0.100	0.179	0.356	0.179
ionosphere	104.860	**0.000**	64.000	**0.000**	103.970	**0.000**	104.690	**0.000**	79.913	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
wbcd	55.357	**0.000**	32.000	**0.000**	51.984	**0.000**	55.330	**0.000**	8.431	**0.000**	10.000	**0.000**	1.000	**0.000**	0.100	**0.000**	0.010	**0.000**
monk 1	323.270	**0.345**	1.000	**0.345**	315.170	**0.345**	321.800	**0.345**	100.000	**0.345**	10.000	**0.345**	1.000	**0.345**	0.736	**0.345**	0.607	**0.345**
monk 2	512.000	**0.327**	512.000	**0.327**	512.000	**0.327**	512.000	**0.327**	100.000	**0.327**	10.000	**0.327**	1.000	**0.327**	0.100	**0.327**	0.010	**0.327**
monk 3	90.586	**0.167**	64.000	**0.167**	90.510	**0.167**	90.510	**0.167**	101.850	**0.167**	10.000	**0.167**	1.000	**0.167**	0.004	**0.167**	0.004	**0.167**
diabete	64.735	**0.266**	64.000	**0.266**	64.000	**0.266**	64.445	**0.266**	48.731	**0.266**	10.143	0.271	1.000	0.271	0.100	0.276	0.010	0.276
hillvalley	0.004	**0.274**	0.004	**0.274**	0.004	**0.274**	0.004	**0.274**	99.964	0.459	1.226	0.409	0.910	0.406	0.101	0.373	0.011	0.310
german	43.454	**0.240**	1.000	0.252	42.224	**0.240**	43.411	**0.240**	30.248	**0.240**	40.438	**0.240**	2.128	0.252	0.100	0.256	0.010	0.256
svmguide3	0.114	**0.143**	0.031	**0.143**	0.109	**0.143**	0.113	**0.143**	98.481	0.209	10.115	0.162	0.883	0.146	0.102	**0.143**	0.026	0.146
splice	195.480	**0.174**	256.000	0.180	194.010	0.175	195.360	**0.174**	100.000	0.179	10.093	0.177	1.000	**0.174**	0.100	0.178	0.010	0.178

Table 6.4: Optimal λ value and Test error rates for GO, GRID-i and GRAD-i of the first realization. The smallest error rate for each data set is highlighted in bold.

Dataset	GO		GRID-1		GRID-0.1		GRID-0.01		GRAD-100		GRAD-10		GRAD-1		GRAD-0.1		GRAD-0.01	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
colon	**0.133**	0.070	**0.133**	0.070	**0.133**	0.070	**0.133**	0.070	0.140	0.080	0.140	0.080	0.140	0.080	0.140	0.080	0.140	0.080
leukemia	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027	**0.017**	0.027
sonar	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000
heart	0.158	0.029	0.160	0.036	0.158	0.032	0.157	0.030	**0.149**	0.046	0.158	0.035	0.155	0.039	0.164	0.025	0.164	0.023
ionosphere	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000
wbcd	0.001	0.002	0.001	0.002	0.001	0.002	0.001	0.002	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000	**0.000**	0.000
monk 1	0.325	0.032	**0.323**	0.034	0.325	0.032	0.325	0.032	0.331	0.030	0.331	0.030	**0.323**	0.034	**0.323**	0.034	**0.323**	0.034
monk 2	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036	**0.361**	0.036
monk 3	0.191	0.047	0.194	0.044	**0.188**	0.046	**0.188**	0.046	0.199	0.043	0.192	0.033	0.198	0.146	0.194	0.044	0.194	0.044
diabete	0.230	0.039	0.288	0.038	0.228	0.042	0.229	0.040	0.231	0.044	0.229	0.040	**0.225**	0.037	**0.225**	0.038	**0.225**	0.038
hillvalley	0.311	0.065	**0.299**	0.070	**0.299**	0.072	0.311	0.065	0.389	0.120	0.316	0.081	0.361	0.070	0.313	0.078	0.317	0.074
german	**0.249**	0.032	0.254	0.035	0.252	0.036	0.250	0.035	0.250	0.032	**0.249**	0.032	0.252	0.036	0.252	0.037	0.252	0.037
svmguide3	**0.203**	0.096	0.204	0.098	**0.203**	0.096	**0.203**	0.096	0.209	0.096	0.207	0.105	0.208	0.089	0.205	0.101	0.208	0.105
splice	**0.158**	0.013	0.159	0.015	0.159	0.015	0.159	0.015	**0.158**	0.015	**0.158**	0.015	**0.158**	0.013	0.159	0.014	0.160	0.014

Table 6.5: Mean and Standard Deviations of $E^{\dagger}$ of GO, GRID-i and GRAD-i over the the 10 realizations. The smallest Mean for each data set is highlighted in bold.

Chapter 7

Conclusions

This concluding chapter outlines the main contributions of this thesis, and points out some potential directions for future works.

7.1 Contributions

Wrapper-based Feature Selection Method for MLP

In Chapter 3, a new wrapper-based feature selection method for MLP is proposed. This method measures the importance of a feature by the sensitivity of the probabilistic output of MLP with/without this feature. In experiment, the proposed method is compared with three benchmark methods reviewed in Chapter 2, FisherS [31], mRMR [53] and MOI [72]. The advantage of the proposed method over the three benchmark methods is evidently illustrated by the following main results:

(1) In the Weston problems with four different settings, the proposed method consistently outperforms FisherS and MtualI in all settings, and performs comparably with MOI in the first setting while outperforms in the rest three settings.

(2) In the three Corral problems, Corral-6, Corral-46 and Corral-47, the proposed method consistently outperforms FisherS and MtualI in all three problems, and performs comparably with MOI in Corral-6 while outperforms in Corral-46 and Corral-47.

(3) In the eight real-world problems, the proposed method consistently performs at least as well, if not better than the three benchmark methods at all levels of feature selection except the case that only top 1 or 2 features are considered.

A paper [4*], listed on Page 153, based on this work has been published.

Wrapper-based Feature Selection Method for SVR

In Chapter 4, a new wrapper-based feature selection method for SVR is proposed. Similar to the method proposed in Chapter 3, this method measures the importance of a feature by the sensitivity of the probabilistic output of SVR with respect to this feature. Numerical experiments on both artificial and real-world problems demonstrates the advantage of the proposed method over five benchmark methods reviewed in Chapter 2, mRMR [53], HSIC [74], $\Delta\|\omega\|^2$ [33], RMB [65], SpanB [65]. Specifically,

(1) In the three artificial problems, each with four settings, the proposed method consistently performs better than all benchmark methods.

(2) In the six real-world problems, the proposed method consistently performs at least as well, if not better than the five benchmark methods at all levels of feature selection except the case that only top 1 or 2 features are considered.

(3) Compared with the similar wrapper method RMB and SpanB, the proposed method can safely reduce the computational cost due to Theorem 4.2.1.

A paper [2*], listed on Page 153, based on this work has been published.

Filter-based Feature Selection Method using Mutual Information

In Chapter 5, a new filter-based feature selection method using mutual information estimation is proposed. Unlike other mutual information based method, the proposed method measures the importance of a feature in a backward feature selection framework considering all features. Numerical experiments show that the proposed method generally outperforms five benchmark methods reviewed in Chapter 2, mRMR [53], Kwak [47] and HSIC [74] according to the following main results:

(1) In the three artificial Monk problems, each with four settings, the proposed method consistently performs better than all benchmark methods.

(2) In the Weston problem with four settings, the proposed method consistently outperforms mRMR and Kwak in all settings, and performs comparably with HSIC in general. The success of the proposed method on this problem and Monk problems shows that the proposed method can effectively handle the interacting effect of features.

(3) In the six real-world problems, the proposed method consistently performs at least as well, if not better than the three benchmark methods at all levels of feature selection except the case that only top 1 or 2 features are considered.

A technique report [5*], listed on Page 154, based on this work has been published.

Finding Global Minimum of Some Common Validation Function in Support Vector Machine

In Chapter 6, a new method to determine the global optima C values of common validation functions for SVM classifier over a validation set or cross-validation set is proposed. To the best of our knowledge, there is no existing methods that can make this achievement. The advantage of the proposed method over benchmark methods reviewed in Chapter 2, grid search method (GRID) and grad based method (GRAD) [45], is validated in numerical experiments on 14 real-world data sets. Specifically,

(1) The proposed method obtains the global minimal cross validation error rate for all 14 data sets. The GRID method does so for 71% to 100% of the data sets while GRAD method does so around 36% to 43% of the data sets.

(2) In the case when there are multiple C values that attains the global optimum of the cross validation function, the smallest C value is returned by the proposed method. This is not so for GRAD and GRID methods.

(3) The proposed method obtains the lowest mean test error rate in 8 of the 14 data sets, and GRID and GRAD can achieve the same performance only with 1 of 3 (for

GRID) and 5 (for GRAD) settings while these two methods with other settings are worse off.

A paper [1*], listed on Page 153, based on this work has been published.

Although feature selection and model selection are different topics, both play the role of the prepossessing procedure for learning algorithm as mentioned before. To the best of our knowledge, in practice these two techniques are often used together in a learning task. In terms of how to choose feature selection methods, it highly depends on the adopted learning algorithm and the requirement of learning tasks.

7.2 Directions of Future Work

Several directions are available for future research based on the work in this thesis.

Feature Selection for Semi-Supervised and Unsupervised Problems

In many applications, labeling input samples is often difficult or time consuming due to the prohibitive effort of experienced human annotators [91]. An alternative is to look into semi-supervised and unsupervised learning paradigms. In these two learning paradigms, a few labeled samples (only for semi-supervised problem) and large amount of unlabeled samples are available. Obviously , traditional supervised feature selection methods are challenged by the situation that the label information is unavailable or rather insufficient. To the best of our knowledge, semi-supervised and unsupervised feature selection methods are still very limited and would benefit from further research in these

direction.

Selecting the Global Optimal Regularization Parameter for Other Variants of SVM

As addressed in Rosset et al. [69], the solution path algorithm [34, 58] can be extended to other classification algorithms, such as logistic regression, 1-norm SVM [90] and least square SVM [76]. Obviously, the proposed model selection approach in Chapter 6 can be easily applied to all these classification algorithms. However, it is not easy to directly apply the proposed method of Chapter 6 to SVR algorithm, although the solution path of SVR on the regularization parameter C and the derivation parameter ε have been proposed in [85, 86]. In regression problems, the validation functions on parameters C and ε are quite different from those in classification problems and careful investigation are needed to extend the method of Chapter 6 to SVR.

Choosing the Global Optimal Kernel for SVM

Kernel parameter is another important hyperparameter in SVM. This has attracted much attention recently, such as non-parametric kernel learning [92, 50, 36, 37], multiple kernel learning [82, 66, 2, 75], the solution path of SVM on kernel parameter [85, 3, 68] and gradient based methods [45, 12]. However, the global optimal kernel for SVM cannot be assured. Efforts in this direction would be helpful.

Bibliography

[1] A.Asuncion and D.J.Newman. UCI machine learning repository, 2007.

[2] F. R. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, November 2008.

[3] F. R. Bach, R. Thibaux, and M. I. Jordan. Computing regularization paths for learning multiple kernels. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 73–80, Cambridge, MA, 2005. MIT Press.

[4] R. Battiti. Using mutual information for selection features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.

[5] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, November 1995.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[7] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal

margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, New York, NY, USA, 1992. ACM.

[8] L. Breiman. Random forests. *Machine Learning*, (45:1):5–32, October 2001.

[9] R. H. Byrd, P. Lu, and J. Nocedal. A limited-memory algorithm for bound-constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995.

[10] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[11] M.-W. Chang and C.-J. Lin. Leave-one-out bounds for support vector regression model selection. *Neural Computation*, 17(5):1188–1222, 2005.

[12] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46:131–159, 2002.

[13] T. W. Chow and D. Huang. Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information. *IEEE Transactions on Neural Networks*, 16(1):1045–9227, January 2005.

[14] W. Chu, S. S. Keerthi, and C. J. Ong. Bayesian support vector regression using a unified loss function. *IEEE Transactions on Neural Networks*, 15:29–44, 2004.

[15] K.-M. Chung, W.-C. Kao, C.-L. Sun, L.-L. Wang, and C.-J. Lin. Radius margin bounds for support vector machines with the RBF kernel. *Neural Computation*, 15(11):2643–2681, 2003.

[16] D. Clark, Z. Schreter, and A. Adams. A quantitative comparison of dystal and backpropagation. In *the Australian Conference on Neural Networks (ACNN'96)*, 1996.

[17] T. M. Cover and J. A. Thomas. *Elements of information theory (2nd Edition)*. Wiley-Interscience, 2006.

[18] G. Cybenko. Continuous valued neural networks with two hidden layers are sufficient. Technical report, Department of Computer Science, Tufts University, Medford, MA,, 1988.

[19] K. Duan, S. S. Keerthi, and A. N. Poo. Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51:41 C 59, 2003.

[20] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, November 2000.

[21] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada. Normalized mutual information feature selection. *IEEE Transactions on Neural Networks*, 20(2):189–201, February 2009.

[22] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification, 2008.

[23] F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.

[24] A. M. Fraser and H. L. Swinney. Independent coordinates for strange attractors from mutual information. *Physical Review A*, 33(2):1134–1140, Feb 1986.

[25] J. H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991.

[26] J. B. Gao, S. R. Gunn, C. J. Harris, and M. Brown. A probabilistic framework for SVM regression and error bar estimation. *Machine Learning*, 46(1-3):71–89, 2002.

[27] M. Girolami and C. He. Probability density estimation from optimally condensed data samples. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 25:1253–1264, 2003.

[28] A. Gretton, O. Bousquet, A. Smola, and B. Schoelkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *ALT 2005*, pages 63–78, 10/08/ 2005.

[29] O. Gualdrón, J. Brezmes, E. Llobet, A. Amari, X. Vilanova, B. Bouchikhi, and X. Correig. Variable selection for support vector machine based multisensor systems. *Sensors and Actuators B: Chemical*, 122:259–268, March 2007.

[30] L. Gunter and J. Zhu. Efficient computation and model selection for the support vector regression. *Neural Computation*, 19:1633–1655, 2007.

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

[32] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors. *Feature Extraction: Foundations and Applications (Studies in Fuzziness and Soft Computing)*. Springer Verlag, August 2006.

[33] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[34] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391 – 1415, October 2004.

[35] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning (2nd Edition)*. Springer, 2009.

[36] S. C. Hoi and R. Jin. Active kernel learning. In *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, 2008.

[37] S. C. Hoi, R. Jin, and M. R. Lyu. Learning non-parametric kernel matrices from pairwise constraints. In *Proceedings of the 24th International Conference on Machine Learning*, OR, US, 2007.

[38] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[39] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *ICML '08: Proceedings of the 25th International Conference on Machine learning*, pages 408–415, New York, NY, USA, 2008. ACM.

[40] C.-N. Hsu, H.-J. Huang, and S. Dietrich. The annigma-wrapper approach to fast

feature selection for neural nets. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 32(2):207–212, 2002.

[41] T. Joachims. *Making large-Scale SVM Learning Practical.*, chapter In B. Scholkopf, C. Burges and A. Smola (Eds), Advances in kernel methods: Support Vector Learning. MIT Press, 1998.

[42] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and subset selection problem. In *International Conference on Machine Learning*, pages 121–129, San mateo.CA, 1994.

[43] J.S.Bridle. *Neurocomputing: Algorithms, Architectures and Applications*, chapter Probabilistic interpretation of feedfoward classification network outputs with relationships to statistical pattern recognition, pages 227–236. Springer-Verlag, 1989.

[44] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13(3):637–649, 2001.

[45] S. S. Keerthi, V. Sindhwani, and O. Chapelle. An efficient method for gradient-based adaptation of hyperparameters in SVM models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 673–680. MIT Press, Cambridge, MA, 2007.

[46] N. Kwak and C.-H. Choi. Input feature selection by mutual information based on

parzen window. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1667–1671, 2002.

[47] N. Kwak and C. H. Choi. Input feature selection for classification problems. *IEEE Transactions on Neural Networks*, 13(1):143 – 159, January 2002.

[48] M. H. Law and J. T. Kwok. Bayesian support vector regression. In *In Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, pages 239–244, 2001.

[49] W. Li. Mutual information functions versus correlation functions. *Journal of Statistical Physics*, 60:823–837, September 1990.

[50] Z. Li, J. Liu, and X. Tang. Pairwise constraint propagation by semidefinite programming for semi-supervised classification. In *Proceedings of the 25th International Conference on Machine learning*, New York, NY, USA, 2008.

[51] C. J. Lin and R. C. Weng. Simple probabilistic predictions for support vector regression. Technical report, Department of Cmputer Science, National Taiwan University, 2004.

[52] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, April 2005.

[53] F. Long, H. Peng, and C. Ding. Feature selection based on mutual information: criteria of max-dependency ,max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, August 2005.

[54] D. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

[55] M. Martin Fodslette. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, 1993.

[56] K. Miyahara and M. J. Pazzani. Collaborative filtering with the simple bayesian classifier. In *In: Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, pages 679–689, 2000.

[57] I. Nabney and C. Bishop. Netlab neural network software.

[58] C.-J. Ong, S.-Y. Shao, and J.-B. Yang. An improved algorithm for the solution of the regularization path of SVM. *IEEE Transactions on Neural Networks*, 21(3):451–462, 2010.

[59] E. S. Page. A note on generating random permutations. *Applied Statistics*, 16(3):273–274, 1967.

[60] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[61] W. Penny. Kullback-Liebler divergences of Normal, Gamma, Dirichlet and Wishart densities. Technical report, Wellcome Department of Cognitive Neurology, University College London, 2001.

[62] J. C. Platt. *Fast training of support vector machines using sequential minimal optimization*, chapter In B. Scholkopf, C. Burges and A. Smola (Eds), Advances in kernel methods: Support Vector Learning. MIT Press, 1998.

[63] J. C. Platt. *Using sparseness and analytic QP to speed training of support vector machines*, chapter In M.S. Kearns, S.A. Solla and D. A. Cohn (Eds), Advances in Neural Information Processing Systems, 11. Cambridge, MIT Press, 1998.

[64] A. Rakotomamonjy. Variable selection using SVM-based criteria. *Journal of Machine Learning Research*, 3:1357–1320, 2003.

[65] A. Rakotomamonjy. Analysis of SVM regression bounds for variable ranking. *Neurocomputing*, 70(7-9):1489 – 1501, 2007.

[66] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, November 2008.

[67] G. Rätsch. Benchmark repository, 2005.

[68] S. Rosset. Following curved regularized optimization solution paths. In *Advances in Neural Information Processing Systems 17*, 2005.

[69] S. Rosset and J. Zhu. Piecewise linear regularized solution paths. *Annals of Statistics*, 35:1012, 2007.

[70] R. Setiono and H. Liu. Neural-network feature selector. *IEEE Transactions on Neural Networks*, 8(3):29–44, 1997.

[71] K.-Q. Shen, C.-J. Ong, X.-P. Li, and E. P. Wilder-Smith. Feature selection via sensitivity analysis of SVM probabilistic outputs. *Machine Learning*, 70(1):1–20, 2008.

[72] V. Sindhwani, S. Rakshit, D. Deodhare, D. Erdogmus, J. Principe, and P. Niyogi. Feature selection in MLPs and SVMs based on maximum output information. *IEEE Transactions on Neural Networks*, 15(4):937–948, July 2004.

[73] A. J. Smola and B. Scholkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[74] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Supervised feature selection via dependence estimation. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 823–830, New York, NY, USA, 2007. ACM.

[75] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, July 2006.

[76] J. Suykens, L. Lukas, P. V. Dooren, B. D. Moor, J. Vandewalle, and U. C. D. Louvain. Least squares support vector machine classifiers: a large scale algorithm. *Neural Processing Letters*, 9(3):293–300, 1999.

[77] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 58(1):267–288, 1996.

[78] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6:1453–1484, September 2005.

[79] V. Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer, November 1995.

[80] V. Vapnik and O. Chapelle. Bounds on Error Expectation for Support Vector Machines. *Neural Computation*, 12(9):2013–2036, 2000.

[81] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998.

[82] M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1065–1072, 2009.

[83] M. Vasconcelos and N. Vasconcelos. Natural image statistics and low-complexity feature selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):228–244, 2009.

[84] A. Verikas and M. Bacauskiene. Feature selection with neural networks. *Pattern Recognition Letters*, 23(11):1323–1335, 2002.

[85] G. Wang, D.-Y. Yeung, and F. H. Lochovsky. A kernel path algorithm for support vector machines. In *ICML '07: Proceedings of the 24th International Conference on Machine learning*, pages 951–958, New York, NY, USA, 2007. ACM.

[86] G. Wang, D.-Y. Yeung, and F. H. Lochovsky. A new solution path algorithm in support vector regression. *IEEE Transactions on Neural Networks*, 19(10):1753–1767, October 2008.

[87] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, pages 668–674, 2000.

[88] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and*

Techniques with Java Implementations (The Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann, 1st edition, October 1999.

[89] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[90] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Neural Information Processing Systems*, page 16. MIT Press, 2003.

[91] X. Zhu. Semi-supervised learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin, Madison, 2005.

[92] J. Zhuang, I. W. Tsang, and S. C. H. Hoi. Simplenpkl: simple non-parametric kernel learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1273–1280, 2009.

Appendices

A. Proof of the Theorem 3.2.1

Proof. Since $x_{(j)}$ is derived from x after the values of the j^{th} feature having been uniformly randomly permuted by the RP process, the distribution of x^j is unchanged, or

$$p(x^j_{(j)}) = p(x^j).$$

Hence, we have

$$p(x_{(j)}) = p(x^j_{(j)}, x_{-j}) = p(x^j_{(j)})p(x_{-j}) = p(x^j)p(x_{-j}),$$

Using similar argument, we have

$$p(x_{(j)}, \omega_k) = p(x^j_{(j)})p(x_{-j}, \omega_k) = p(x^j)p(x_{-j}, \omega_k).$$

Hence,

$$p(\omega_k|x_{(j)}) = \frac{p(x_{(j)}, \omega_k)}{p(x_{(j)})} = \frac{p(x^j)p(x_{-j}, \omega_k)}{p(x^j)p(x_{-j})} = p(\omega_k|x_{-j}).$$

□

B. KL Divergence of Two Laplace Distributions

This appendix shows the explicit expression of $D_{KL}(p_1(x); p_2(x))$ when $p_1(x)$ and $p_2(x)$ are Laplace distributions. For convenience, let $y := \mu_1 - x$ and $\theta := |\mu_1 - \mu_2|$. Then,

$$p_1(x) = \frac{1}{2\sigma_1}\exp(-\frac{|\mu_1 - x|}{\sigma_1}) \Leftrightarrow p_1(y) = \frac{1}{2\sigma_1}\exp(-\frac{|y|}{\sigma_1}),$$

$$p_2(x) = \frac{1}{2\sigma_2}\exp(-\frac{|\mu_2 - x|}{\sigma_2}) \Leftrightarrow p_2(y) = \frac{1}{2\sigma_2}\exp(-\frac{|\theta \pm y|}{\sigma_2}).$$

Using them,

$$\begin{aligned}
&D_{KL}(p_1(y); p_2(y)) \\
&= \int_{-\infty}^{\infty} \frac{1}{2\sigma_1}\exp(-\frac{|y|}{\sigma_1}) \ln \frac{\frac{1}{2\sigma_1}\exp(-\frac{|y|}{\sigma_1})}{\frac{1}{2\sigma_2}\exp(-\frac{|\theta \pm y|}{\sigma_2})} dy \\
&= \int_{-\infty}^{\infty} \frac{1}{2\sigma_1}\exp(-\frac{|y|}{\sigma_1})[\ln \frac{\sigma_2}{\sigma_1} - \frac{|y|}{\sigma_1} + \frac{|\theta \pm y|}{\sigma_2}] dy \\
&= \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} \int_{-\infty}^{\infty} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} - \frac{1}{2}\int_{-\infty}^{\infty} \frac{|y|}{\sigma_1}\exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2}\int_{-\infty}^{\infty} \frac{|\theta \pm y|}{\sigma_1}\exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1}
\end{aligned}$$

Case 1: Suppose $|\theta \pm y| = |\theta + y|$. Expression $D_{KL}(p_1(y); p_2(y))$ becomes

$$
\begin{aligned}
&= \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} \int_{-\infty}^{\infty} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} - \frac{1}{2} \int_{-\infty}^{\infty} \frac{|y|}{\sigma_1} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2} \int_{-\infty}^{\infty} \frac{|\theta + y|}{\sigma_1} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} \\
&= \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} \int_{-\infty}^{0} \exp(\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} \int_{0}^{\infty} \exp(-\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} - \frac{1}{2} \int_{-\infty}^{0} \frac{-y}{\sigma_1} \exp(\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} \\
&\quad - \frac{1}{2} \int_{0}^{\infty} \frac{y}{\sigma_1} \exp(-\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2} \int_{-\infty}^{-\theta} \frac{-\theta - y}{\sigma_1} \exp(\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} \\
&\quad + \frac{\sigma_1}{2\sigma_2} \int_{-\theta}^{0} \frac{\theta + y}{\sigma_1} \exp(\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2} \int_{0}^{\infty} \frac{\theta + y}{\sigma_1} \exp(-\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} \\
&= \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} + \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} - \frac{1}{2} - \frac{1}{2} + \frac{\sigma_1}{2\sigma_2} \exp(\frac{-\theta}{\sigma_1}) + \frac{\sigma_1}{2\sigma_2} (\frac{\theta}{\sigma_1} - 1 + \exp(\frac{-\theta}{\sigma_1})) + \frac{\sigma_1}{2\sigma_2} (\frac{\theta}{\sigma_1} + 1) \\
&= \ln \frac{\sigma_2}{\sigma_1} - 1 + \frac{\sigma_1}{\sigma_2} \exp(-\frac{\theta}{\sigma_1}) + \frac{\theta}{\sigma_2}
\end{aligned}
$$

Case 2: Alternatively, if $|\theta \pm y| = |\theta - y|$, expression $D_{KL}(p_1(y); p_2(y))$ becomes

$$
\begin{aligned}
&= \frac{\ln \frac{\sigma_2}{\sigma_1}}{2} \int_{-\infty}^{\infty} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} - \frac{1}{2} \int_{-\infty}^{\infty} \frac{|y|}{\sigma_1} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2} \int_{-\infty}^{\infty} \frac{|\theta - y|}{\sigma_1} \exp(-\frac{|y|}{\sigma_1}) d\frac{y}{\sigma_1} \\
&= \ln \frac{\sigma_2}{\sigma_1} - 1 + \frac{\sigma_1}{2\sigma_2} \int_{-\infty}^{0} \frac{\theta - y}{\sigma_1} \exp(\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} + \frac{\sigma_1}{2\sigma_2} \int_{0}^{\theta} \frac{\theta - y}{\sigma_1} \exp(-\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} \\
&\quad + \frac{\sigma_1}{2\sigma_2} \int_{\theta}^{\infty} \frac{y - \theta}{\sigma_1} \exp(-\frac{y}{\sigma_1}) d\frac{y}{\sigma_1} \\
&= \ln \frac{\sigma_2}{\sigma_1} - 1 + \frac{\theta}{\sigma_2} + \frac{\sigma_1}{\sigma_2} \exp(-\frac{\theta}{\sigma_1})
\end{aligned}
$$

C. Gradient-based Model Selection

Gradient-based hyperparameters tuning method for SVM proposed by Keerthi et al. [45] requires a continuously differentiable function with respect to λ. Using the notations of this paper, the approximation proposed in [45] for $Err(\lambda)$ function of (6.19) is

$$\widetilde{Err}(\lambda) = 1 - \frac{1}{|\mathscr{I}_V|} \sum_{j \in \mathscr{I}_V} s_j = 1 - \frac{1}{|\mathscr{I}_V|} \sum_{j \in \mathscr{I}_V} \frac{1}{1 + \exp(-\rho(\lambda) y_j h_j(\lambda))}$$

with $\rho(\lambda) := \frac{10}{\sqrt{\frac{1}{|\mathscr{I}_V|} \sum_{i \in \mathscr{I}_V} (h_i(\lambda) - \bar{h}(\lambda))^2}}$ and $\bar{h}(\lambda) = \frac{1}{|\mathscr{I}_V|} \sum_{i \in \mathscr{I}_V} h_i(\lambda)$. The expression of its gradient is

$$\frac{d\widetilde{Err}(\lambda)}{d\lambda} = \sum_{j \in \mathscr{I}_V} \frac{\partial \widetilde{Err}}{\partial s_j} \frac{\partial s_j}{\partial \lambda} = \sum_{j \in \mathscr{I}_V} \frac{\partial \widetilde{Err}}{\partial s_j} \left[\frac{\partial s_j}{\partial \rho} \left(\sum_{i \in \mathscr{I}_V} \frac{\partial \rho}{\partial h_i} \frac{\partial h_i}{\partial \lambda} \right) + \frac{\partial s_j}{\partial h_j} \frac{\partial h_j}{\partial \lambda} \right]$$

with

$$\begin{aligned}
\frac{\partial \widetilde{Err}}{\partial s_j} &= -\frac{1}{|\mathscr{I}_V|} \\
\frac{\partial s_j}{\partial \rho} &= s_j(1 - s_j) y_j h_j \\
\frac{\partial s_j}{\partial h_j} &= s_j(1 - s_j) \rho(\lambda) y_j, \\
\frac{\partial \rho}{\partial h_i} &= -\frac{10(h_i(\lambda) - \bar{h}(\lambda))}{|\mathscr{I}_V| \rho^3(\lambda)}
\end{aligned}$$

and

$$\frac{\partial h_j}{\partial \lambda} = \frac{h_j^{\ell+1} - h_j^{\ell}}{\lambda^{\ell+1} - \lambda^{\ell}}.$$

Note that these expressions are based on (6.21) and the development of this paper. In the case where the regularization solution path is not available, a different set of expressions is needed. In particular, $\frac{\partial h_j}{\partial \lambda}$ requires the inverse of an appropriate matrix obtained using data points in $\mathscr{E}(\lambda)$ and constraint $\sum_i \alpha_i y_i = 0$, see [45] for details.

Author's Publications

Journal Papers

[1*] **Jian-Bo Yang** and Chong-Jin Ong. "Determination of Global Minima of Some Common Validation Functions in Support Vector Machine," *IEEE Transactions on Neural Network,* vol. 22, no. 4, pp. 654 - 659, 2011.

[2*] **Jian-Bo Yang** and Chong-Jin Ong. "Feature Selection using Probabilistic Prediction of Support Vector Regression," *IEEE Transactions on Neural Network,* vol. 22, no. 6, pp. 954 - 962, 2011.

[3*] Chong-Jin Ong, Shi-Yun Shao and **Jian-Bo Yang**. "An Improved Algorithm for the Solution of the Regularization Path of Support Vector Machine," *IEEE Transactions on Neural Network,* vol. 21, no. 3, pp. 451 - 462, 2010.

[4*] **Jian-Bo Yang**, Kai-Quan Shen, Chong-Jin Ong, and Xiao-Ping Li. "Feature Selection for MLP Neural Network: The Use of Random Permutation of Probabilistic

Outputs," *IEEE Transactions on Neural Network,* vol. 20, no. 12, pp. 1911 - 1922, 2009.

Technical Report

[5*] **Jian-Bo Yang** and Chong-Jin Ong. "Feature Selection via Estimation of High-Dimensional Mutual Information," *Technical Report C11-001, Dept. of Mechanical Engineering, 2011*.

Conference Papers

[6*] **Jian-Bo Yang** and Chong-Jin Ong. "Feature Selection for Support Vector Regression Using Probabilistic Prediction," *16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 343-352, 2010.

[7*] **Jian-Bo Yang**, Kai-Quan Shen, Chong-Jin Ong, and Xiao-Ping Li. "Feature selection via sensitivity analysis of MLP probabilistic outputs," *2008 IEEE International Conference on Systems, Man and Cybernetics*, pp. 774 - 779, 2008.