UNIVERSITY OF SOUTHAMPTON

Comparison and Performance Enhancement of Modern Pattern Classifiers

by

Somjet Suppharangsan

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

November 2010

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Somjet Suppharangsan

This thesis is a critical empirical study, using a range of benchmark datasets, on the performance of some modern machine learning systems and possible enhancements to them. When new algorithms and their performance are reported in the machine learning literature, most authors pay little attention to reporting the statistical significances in performance differences. We take Gaussian process classifiers as an example, which shows disappointing number of performance evaluations in the literature. What is particularly ignored is any use of the uncertainties in the performance measures when making comparisons. This thesis makes a novel contribution by developing a methodology for formal comparisons that also include performance uncertainties. Using support vector machine (SVM) as classification architectures, the thesis explores two potential enhancements to complexity reduction: (a) subset selection on the training data by some pre-processing approaches, and (b) organising the classes of a multi-class problem in a tree structure for fast classification. The former is crucial, as dataset sizes are known to have increased rapidly, and the straightforward training using quadratic programming over all of the given data is prohibitively expensive. While some researchers focus on training algorithms that operate in a stochastic manner, we explore data reduction by cluster analysis. Multi-class problems in which the number of classes is very large are of increasing interest. Our contribution is to speed up the training by removing as many irrelevant data as possible and preserving the potential data that are believed to be support vectors. The results show that too high a data reduction rate can degrade performance. However, on a subset of problems, the proposed methods have produced comparable results to the full SVM despite the high reduction rate. The new learning tree structure can then be combined with the data selection methods to obtain a further increase in speed. Finally, we also critically review SVM classification problems in which the input data is binary. In the chemoinformatics and bioinformatics literature, the Tanimoto kernel has been empirically shown to have good performance. The work we present, using carefully set up synthetic data of varying dimensions and dataset sizes, casts doubt on such claims. Improvements are noticeable, but not to the extent claimed in previous studies.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Name
AUC	Area Under the (ROC) Curve
CV	Cross-Validation
DAG	Directed Acyclic Graph
ECOC	Error Correcting Output Code
EP	Expectation Propagation
FITC	Fully Independent Training Conditional approximation
FLDA	Fisher's Linear Discriminant Analysis
GP	Gaussian Process
IVM	Informative Vector Machine
KDE	Kernel Density Estimation
KKT	Karush-Kuhn-Tucker conditions
KL	Kullback-Leibler divergence
LDA	Linear Discriminant Analysis
MCMC	Markov Chain Monte Carlo
OVA	One-versus-All
OVO	One-versus-One
QDA	Quadratic Discriminant Analysis
QP	Quadratic Programming
RBF	Radial Basis Function
ROC	Receiver Operating Characteristic
SPGPC	Sparse Pseudo-input Gaussian Process Classification
SVM	Support Vector Machine
UDT	Unbalanced Decision Tree
df	degrees of freedom
sv	support vector

Nomenclature

Symbol	Meaning
α	significance level in hypothesis testing
α_i	support vector corresponding to $\boldsymbol{x}_i$ in SVM
$\approx$	approximated by
$\sim$	distributed according to
$\bar{A}$	average score or average rank of A
$\bigcup_k A_k$	union of all sets A_k, i.e., $A_1 \cup A_2 \cup \ldots \cup A_k$
$\lvert\boldsymbol{C}\rvert$	determinant of $\boldsymbol{C}$
$\lvert a\rvert$	absolute value of a
$\lVert\boldsymbol{x}\rVert$	norm of $\boldsymbol{x}$, i.e., $\sqrt{\sum_i^d \boldsymbol{x}^2}$
$\mathbf{0}$	vector of all 0's
$\mathbf{1}$	vector of all 1's
$\boldsymbol{\mu}$	vector of mean
$\boldsymbol{\Phi}(\boldsymbol{x}_i)$	feature mapping of $\boldsymbol{x}_i$
$\boldsymbol{\Sigma}$	covariance matrix
$\boldsymbol{\Sigma}^{-1}$	inverse of $\boldsymbol{\Sigma}$
$\boldsymbol{\theta}$	vector of hyperparameters
$\boldsymbol{C}$	positive definite covariance matrix in Gaussian process
$\boldsymbol{c}(\boldsymbol{x}_*)$	vector of covariances between latent function $f(\boldsymbol{x}_*)$ and $\boldsymbol{f}$
$\boldsymbol{f}$	Gaussian process latent function values, $\boldsymbol{f} = [f(\boldsymbol{x}_1),\ f(\boldsymbol{x}_2),\ \ldots, f(\boldsymbol{x}_N)\ ]^T$
$\boldsymbol{I}$	identity matrix
$\boldsymbol{K}$	kernel matrix
$\boldsymbol{w}$	weight vector
$\boldsymbol{X}$	input matrix, $\boldsymbol{X} = [\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ \ldots,\ \boldsymbol{x}_N]^T$
$\boldsymbol{X}^T$	transpose of $\boldsymbol{X}$
$\boldsymbol{x}_*$	test data
$\boldsymbol{x}_i$	the ith training data
$\boldsymbol{y}$	vector of y_i, $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^T$
ΔH	differential entropy
$\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$ or $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$	dot product between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$
$\log z$ or $\ln z$	natural logarithm of z (base e)
$\log_J z$	logarithm of z to the base J

Symbol	Meaning
μ	mean
$\odot$	element-wise product of two vectors
$\Phi(z)$	cumulative unit Gaussian: $\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp\left(-\frac{t^2}{2}\right) dt$
σ	standard deviation
$\sigma(z)$	logistic function: $\sigma(z) = \dfrac{1}{1+\exp(-z)}$
σ^2	variance
$\mathrm{diag}(\boldsymbol{W})$	vector containing the diagonal elements of matrix $\boldsymbol{W}$
$\mathrm{diag}(\boldsymbol{w})$	diagonal matrix containing the elements of vector $\boldsymbol{w}$
$\bigtriangledown$	partial derivatives
$\bigtriangledown\bigtriangledown$	the Hessian matrix of second derivatives
ξ_i	the slack variable which measuring the degree of misclassification of the ith data
C	SVM trade-off parameter
$c(\boldsymbol{x}_*, \boldsymbol{x}_*)$	variance of $f(\boldsymbol{x}_*)$
$cov(\boldsymbol{X})$	a MATLAB function computing the covariance of $\boldsymbol{X}$
D or d	the number of dimensions or features in the input space
F	the number of dimensions or features in the feature space
f_*	Gaussian process prediction of test data: $f_* = f(\boldsymbol{x}_*)$
k or K	the number of classes or the number of clusters
M or m	the number of selected data
$mean(\boldsymbol{X})$	a MATLAB function computing the mean of $\boldsymbol{X}$
N or n	the number of training data
y_*	label of test data
y_i	the ith training label or target
$\mathcal{D}$	training dataset: $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, N\}$
$\mathcal{N}(\boldsymbol{f} \mid \boldsymbol{0}, \boldsymbol{C})$	variable $\boldsymbol{f}$ has a Gaussian (Normal) distribution with mean vector $\boldsymbol{0}$ and covariance matrix $\boldsymbol{C}$ interchangeably written as $\mathcal{N}(\boldsymbol{f}; \boldsymbol{0}, \boldsymbol{C})$
$\mathcal{N}(\boldsymbol{x})$	unit Gaussian $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
$y\|x$ and $p(y\|x)$	conditional random variable y given x and its density distribution
$KL(p(x)\|q(x))$	Kullback-Leibler divergence: $KL(p(x)\|q(x)) = \int p(x) \log \frac{p(x)}{q(x)} dx$

Note that vectors are in lowercase bold type while matrices are bolded and capitalized.

DECLARATION OF AUTHORSHIP

I, Somjet Suppharangsan, declare that the thesis entitled `Comparison and Performance Enhancement of Modern Pattern Classifiers` and the work presented in it are my own. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as:
  - Ramanan, A., Suppharangsan, S. and Niranjan, M. (2007) *Unbalanced Decision Trees for Multi-Class Classification*, In Proceedings of the 2nd IEEE International Conference on Industrial and Information Systems, pages 291–294.

Signed:.....
Date:.....

Acknowledgements

There are many people without whom this thesis would not have been possible. I would like to acknowledge their assistance and support in many aspects. First, I would like to thank my supervisor, Professor Mahesan Niranjan, for his generosity, guidance and encouragement from the initial to the final stage of this work. I also would like to express my gratitude to Dr. Guido Sanguinetti for his useful suggestions and to Dr. Salih Tuna for providing me with the bioinformatics and chemoinformatics datasets.

I am grateful to all of my colleagues and friends both in Sheffield and Southampton, especially Amirthalingam Ramanan, Bassam Farran, Wei Liu, Ke Yuan and Sutat Sae-Tang, for their kindness and friendship. I would like to thank John Wynn for his help when I experienced difficulty with the computer system. I also wish to acknowledge the university library and IRIDIS, a high-performance computer cluster facility. Furthermore, I would like to express my gratitude to the Royal Thai Government for the financial support and to everyone in the Office of Educational Affairs, Royal Thai Embassy in London for the scholarship and passport.

Many other people were involved and I greatly appreciate their help, although all of their names are not shown here. Finally, I am deeply indebted to my parents and my wife for their love. Without their encouragement and support, it would have been impossible to complete this thesis.

Somjet Suppharangsan
Southampton
November 2010

This thesis is dedicated to my grandmother, R-Ma.

Chapter 1

Outline of the Thesis

1.1 Introduction

Classification, particularly the multi-class classification problem, is one of the most important topics in the machine learning community in which many available algorithms are provided. The neural network was one of the most famous algorithms, while recently two state-of-the-art algorithms: Gaussian process (GP) and support vector machine (SVM) have been developed and used widely for the classification problem for at least a decade. They offer a comparable or even better performance than the neural network; however, they suffer from scalability due to their $\mathcal{O}(N^3)$ complexity in which N is the number of training instances. Many researchers have been working on the GP scalable issue, whereas the scalable issue of SVM has not received much attention compared to the former algorithm. Although several methods exist to deal with SVM scalability, the magnitude of the input is still large in contrast to the counterpart, the sparse GP, which provides a much smaller input size.

In theory, the SVM requires a smaller number of input instances, referred to as support vectors, in order to create a classifier to solve the classification problem. Unfortunately, the only way to extract the support vectors correctly is by feeding the entire input into an expensive computation known as quadratic programming (QP). In this thesis, we propose a number of schemes to alleviate this bottleneck issue. The schemes do not, however, aim at improving accuracy, but primarily seeking potential input instances or candidate support vectors that would reduce the cost of resolving QP without seeking more complex mathematical frameworks, as the counterpart GP has done for the sparse GP. In order to make a comparison amongst those schemes and the full SVM, we employ the hypothesis testing. However, traditional hypothesis testing may be not appropriate for use in the experiments due to uncertainty in the cross-validation. The uncertainty is usually ignored by many experimenters, thus possibly invalidating their conclusions.

Therefore, we also propose a new form of statistical hypothesis testing, considering the uncertainty embedded in the cross-validation.

1.2 Thesis Contribution and Outline

The main contribution of this thesis is a new proposed evaluation of the statistical comparison of classifiers' performance, which also considers the uncertainty from the cross-validation. In addition, the investigation of the proposed data selection approaches is presented, and a new version of a tree learning structure is developed for SVM in large multi-class classification problems. The outline of the thesis is as follows:

- Chapter 2 describes the background knowledge of the classification problem, starting with a general binary classification problem and later extending to the multi-class problem. This chapter explains the concept of the support vector machine and Gaussian process, and introduces a new tree learning structure for the multi-class problem.
- Chapter 3 reviews the statistical hypothesis testing for the comparison of the classifiers' performance. Both parametric and non-parametric statistical tests are described. We also point out why traditional statistical hypothesis testing may not be suitable in the classification context in this chapter.
- Chapter 4 analyses the performance of the Gaussian process and compares its performance with non-Gaussian process classifiers in the literature. We will show that there is inconsistency in the literature for making a proper comparison and then experiments will be performed in order to make a formal comparison between the selected classifiers in Chapter 5.
- Chapter 5 presents the proposed hypothesis testing, which takes uncertainty into account. We compare our proposed hypothesis testing with traditional hypothesis testing on a number of classifiers and benchmark datasets. This study will demonstrate in which case the proposed hypothesis testing may be more appropriate.
- Chapter 6 investigates the proposed data selection methods for SVM and shows their results. We introduce an improvement of the new tree learning structure that we described in Chapter 2 for speeding the training. We provide a critical evaluation of the Tanimoto kernel, which is claimed to perform better than the linear kernel on binary data. However, we argue that they are comparable provided that suitable parameters are used.
- Chapter 7 summarises the work, including the limitations of the proposed methods and presents some prospective future work.

Chapter 2

Background Knowledge

This chapter provides basic knowledge relating to classification problems starting with a binary or two-class problem and later shows how to extend the binary solution to a multi-class problem. Consider a set of training data ($\mathcal{D}$) obtained through observations divided into two classes $\{-1, +1\}$:

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), (\boldsymbol{x}_3, y_3), \ldots, (\boldsymbol{x}_N, y_N)\} = \{(\boldsymbol{x}_i, y_i) | i = 1, \ldots, N\}$$

where $\boldsymbol{x}_i$ is the ith data point $\in \mathcal{R}^d$ and y_i is the corresponding class label $\in \{-1, +1\}$. The goal is to create a classifier from the given training dataset $\mathcal{D}$ that minimises misclassification errors on an unseen data set called a test dataset. In other words, the classifier performs a mapping from the test data $\boldsymbol{x}_*$ to predicted label y_*, where the mapping is learnt from the training data. To solve the above classification problem, algorithms such as multi layer perceptron (MLP), k-nearest neighbor (k-nn), and Naive Bayes could be used; however, these algorithms are confronted with local minima and overfitting problems, which depend greatly on initialisation, and are time-consuming (Tran et al., 2005). An algorithm is said to generate the overfitting problem when it yields a low error rate on the training dataset but a higher error rate on the test dataset (Wang and Zhong, 2003).

Two state-of-the-art algorithms for classification, namely, the support vector machine (SVM) and Gaussian process (GP) are described, since their performances offer excellent generalisation. SVM is an example of a classifier that has been increasingly implemented in a number of research works for data mining tasks such as classification, regression, and novelty detection (Bennett and Campbell, 2000). It is an automated supervised learning with high performance compared to many other classifiers (Xiangrong and Fang, 2002). Given its high performance, SVM's concept and derivation are presented in the next sections. Subsequently, the concept of Gaussian process classification is presented, including its approximations. Compared to SVM, which is a determinative approach constructing a decision boundary, GP is attractive, as its probabilistic approach not

only classifies data but also calculates the uncertainty of the output at the same time. In addition to SVM and GP, Fisher's linear discriminant analysis (FLDA) is briefly described due to its simplicity and ability to perform well in separable cases. These three algorithms were applied to a number of datasets in the experiments and evaluated by statistical tests described in a later chapter. For completeness of the underlying classification background, the last section describes a number of methods used to measure classifier performance.

2.1 Support Vector Machine

The foundation of SVM was invented and developed by Vladimir Vapnik and his group at AT&T Bell Laboratories (Vapnik, 1995). The basic idea behind SVM is to determine an optimal hyperplane that separates the training data into two classes with the maximum distance, called maximum margin, between these two classes. Data with the same class label are on one side of the hyperplane, whereas the other class data are on the other side of the hyperplane. The data closest to the hyperplane are called support vectors. Therefore, the optimal hyperplane should maximise the margin and minimise the bound on the generalisation error. The maximum margin hyperplane, shown in Figure 2.1, is

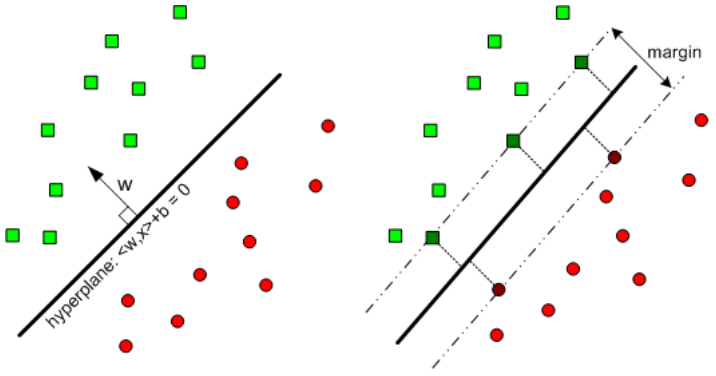


FIGURE 2.1: Maximum margin hyperplane: This graph illustrates the concept of maximum margin hyperplane separating positive examples (green square) from negative examples (red circle); the darker green squares and red circles depict associated support vectors.

$\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0$ where $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ is the dot product between the weight vector $\boldsymbol{w}$ and $\boldsymbol{x}$, or written as $\boldsymbol{w} \cdot \boldsymbol{x}$ in short, and b is a threshold or offset, sometimes referred to as a bias term. The distance between the hyperplane and the nearest $\boldsymbol{x}_i$ is $\frac{|\boldsymbol{w} \cdot \boldsymbol{x}_i + b|}{\|\boldsymbol{w}\|} = \frac{1}{\|\boldsymbol{w}\|}$; as a result, the margin (γ) is $\frac{2}{\|\boldsymbol{w}\|}$. The aim of SVM is to obtain the maximum margin hyperplane; in other words, $\boldsymbol{w}$ and b need to be found so that $\gamma = \frac{2}{\|\boldsymbol{w}\|}$ is maximised. The objective of margin maximisation and associated constraints can be reformulated to the following quadratic programming (QP) as maximising is equivalent to minimising

the reciprocal:

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{2.1}$$

$$\text{subject to} \quad y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geqslant 1 \quad ,\forall i. \tag{2.2}$$

To solve this QP, the Lagrange multiplier method is applied and the Lagrangian function is given by:

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left\{y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) - 1\right\} \tag{2.3}$$

where α_i are Lagrange multipliers and $\alpha_i \geqslant 0$. Setting the derivatives of $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ with respect to $\boldsymbol{w}$ and b then results in:

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \tag{2.4}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0. \tag{2.5}$$

Substituting equation (2.4) and (2.5) for (2.3) gives the following dual representation of the QP problem:

$$\max_{\boldsymbol{\alpha}} \; W(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j \boldsymbol{x}_i \cdot \boldsymbol{x}_j \tag{2.6}$$

subject to constraints:

$$\alpha_i \geqslant 0, \qquad i = 1, \ldots, N \tag{2.7}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0. \tag{2.8}$$

Hence, the solution can be linearly formulated as (2.9). Substituting (2.4) will give the solution (2.10) subject to *Karush-Kuhn-Tucker* (KKT) conditions (2.11)-(2.13), where $\alpha_i > 0$ correspond with support vectors (sv):

$$f(\boldsymbol{x}, \boldsymbol{\alpha}, b) = \boldsymbol{w}\cdot\boldsymbol{x} + b \tag{2.9}$$

$$f(\boldsymbol{x}, \boldsymbol{\alpha}, b) = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i \cdot \boldsymbol{x} + b$$

$$= \sum_{i\in sv} \alpha_i y_i \boldsymbol{x}_i \cdot \boldsymbol{x} + b \tag{2.10}$$

$$\alpha_i \geqslant 0 \tag{2.11}$$

$$y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) - 1 \geqslant 0 \tag{2.12}$$

$$\alpha_i\{y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i) + b) - 1\} = 0 \tag{2.13}$$

Given that directly solving equation (2.1) is more complex, the concept of duality or a dual problem is used to solve alternatively, which also provides the same solution form. The benefit of dual form, which will be seen later, is applying "*kernel trick*" to the product of two vectors. The solution for the new test data $\boldsymbol{x}_*$ is in the form of:

$$f(\boldsymbol{x}_*) = \text{sign}(\sum_{i \in sv} y_i \alpha_i \boldsymbol{x}_i \cdot \boldsymbol{x}_* + b) \tag{2.14}$$

where $\text{sign}(z) = +1$ if $z \geqslant 0$; otherwise $\text{sign}(z) = -1$ and b is calculated by:

$$b = -\frac{max_{y_i=-1}(\boldsymbol{w} \cdot \boldsymbol{x}_i) + min_{y_i=+1}(\boldsymbol{w} \cdot \boldsymbol{x}_i)}{2}. \tag{2.15}$$

The training data $\boldsymbol{x}_i$ corresponding to the parameter $\alpha_i > 0$ are called support vectors, which are the most informative input data for SVM. Removing other data does not affect the solution of SVM. The above derivation essentially is for the case of linearly separable or non-overlapping between classes; however, there are many cases in which some data stand on the wrong side of the hyperplane, as shown in Figure 2.2. To minimise misclassification, the objective function and constraints need to be relaxed by introducing the relaxation factor (Cortes and Vapnik, 1995) $\xi_i \geqslant 0$ and a penalty function $F(\xi) = \sum_{i=1}^{n} \xi_i$, where ξ_i are a measure of the misclassification errors. The modified objective function and its constraints are represented by (2.16)-(2.18), where C is a user-defined regularisation parameter, which is the trade-off between training error and margin. Hence, it is referred to as the trade-off parameter.

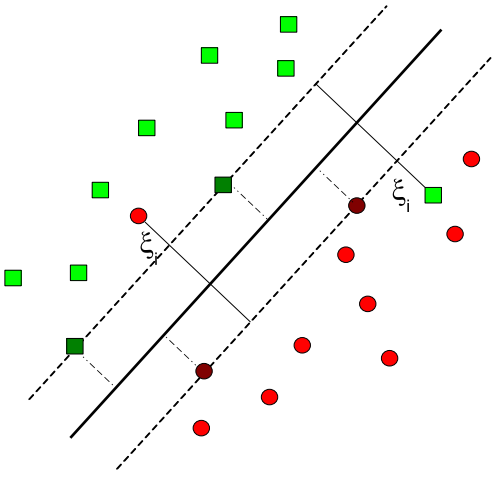


FIGURE 2.2: Linearly non-separable hyperplane

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{2.16}$$

$$\text{subject to :} \quad y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i + b) \geqslant 1-\xi_i \quad ,\forall i \tag{2.17}$$

$$\xi_i \geqslant 0 \quad ,\forall i \tag{2.18}$$

The problem now is to maximise the margin and minimise the total classification error. Similar to the linearly separable case, the quadratic programming or transformation to its dual problem is required to solve (2.16). The solution is identical to the linearly separable case, $f(\boldsymbol{x}_*) = \text{sign}(\sum_{i\in sv} y_i\alpha_i\boldsymbol{x}_i\cdot\boldsymbol{x}_* + b)$, except that b is obtained by the KKT conditions: $\alpha_i(y_i(\boldsymbol{w}\cdot\boldsymbol{x}_i+b)-1+\xi_i)=0$ and $0\leqslant\alpha_i\leqslant C, \forall i$.

Nonetheless, in practice, the classification problems are non-linearly separable. In the non-linearly separable case, SVM transforms data from the input space into a higher dimensional feature space by mapping function $\boldsymbol{\Phi}(\boldsymbol{x}_i) : \mathcal{R}^d \rightarrow \mathcal{R}^F$, as illustrated in Figure 2.3. The corresponding dual formulation becomes:

$$\begin{aligned}\tilde{W}(\boldsymbol{\alpha}) &= \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{\Phi}(\boldsymbol{x}_i)\cdot\boldsymbol{\Phi}(\boldsymbol{x}_j) \\ &= \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)\end{aligned} \tag{2.19}$$

subject to constraints:

$$0 \leqslant \alpha_i \leqslant C \quad ,\forall i \tag{2.20}$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0 \tag{2.21}$$

where $\boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \boldsymbol{\Phi}(\boldsymbol{x}_i)\cdot\boldsymbol{\Phi}(\boldsymbol{x}_j)$ is known as a kernel function. The kernel-based transformation is applied to the non-linearly separable case by changing the inner product or dot product of the mapping function to the kernel function. Thus, the solution of the non-linearly separable case has the form:

$$f(\boldsymbol{x}_*) = \text{sign}(\sum_{i\in sv} y_i\alpha_i\boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_*) + b). \tag{2.22}$$

The kernel function should be easily computed and should satisfy Mercer's condition (Cortes and Vapnik, 1995; Lin and Lin, 2003). Mercer's condition (Vapnik, 1998) states that a mapping $\boldsymbol{\Phi}$ and expansion $\boldsymbol{K}(\boldsymbol{x},\boldsymbol{z}) = \sum_i \Phi_i(\boldsymbol{x})\,\Phi_i(\boldsymbol{z})$ exists only if for any $g(\boldsymbol{x})$ such that $\int g(\boldsymbol{x})^2\,\mathrm{d}\boldsymbol{x} < \infty$ and $\int \boldsymbol{K}(\boldsymbol{x},\boldsymbol{z})g(\boldsymbol{x})g(\boldsymbol{z})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{z} \geqslant 0$. Table 2.1 shows some well-known kernel functions that can be employed in SVM. The advantage of the kernel representation is that there is no need to perform $\boldsymbol{\Phi}$ explicitly because the dot product of those two mappings in the feature space, which is possibly are infinite dimensionality,

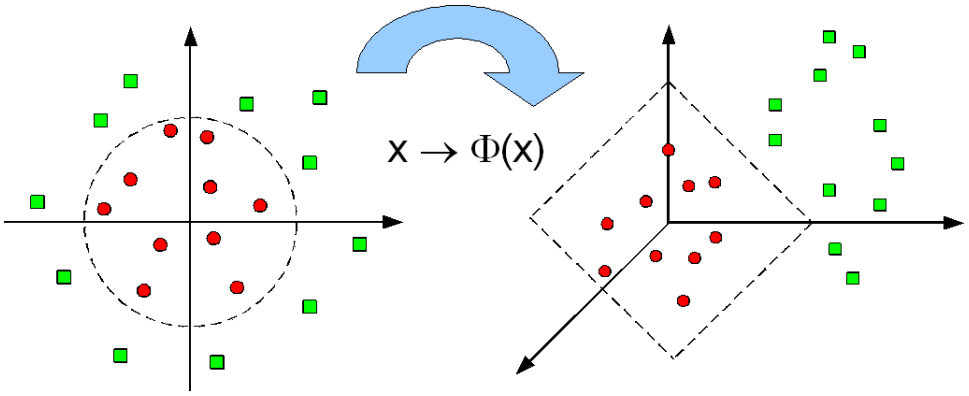


FIGURE 2.3: Mapping input vectors to a higher dimensional space. 2D input space on the left is mapped to 3D feature space on the right.

TABLE 2.1: Examples of the kernel function

Kernel function	Mathematical form $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i)$
Polynomial with degree p	$(\boldsymbol{x} \cdot \boldsymbol{x}_i + 1)^p$
Gaussian Radial Basis Function	$\exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}_i\|^2)$
Sigmoid	$\tanh(\kappa(\boldsymbol{x} \cdot \boldsymbol{x}_i) + \mu)$
Exponential Radial Basis Function	$\exp(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|}{2\sigma^2})$

is now calculated in the input space via the kernel function. However, the optimal choice for the kernel and its parameters are crucial for achieving a good performance. It depends on the user's experience, domain of the problem, and cost of the parameter search. In this thesis, RBF is selected to be used for SVM due to its popularity and promising performance. The RBF kernel is dependent on the norm of the difference between two inputs: one is the support vector and the other is the test data. It has a kernel parameter called γ controlling the smoothness of boundary decision, in which the smaller γ is, the smoother decision surface. The γ parameter determines the area influence of the support vector which behaves as the centre of the RBF kernel over the data space. The mathematical form of the RBF kernel is given in the Table 2.1. Because its form is similar to the Gaussian distribution form, in which we can regard $\gamma = \frac{1}{2\sigma}$, it is also referred to as the Gaussian RBF. As a result, there are two parameters for SVM using RBF kernel, that is, the trade-off parameter C and the kernel parameter γ. However, the parameters are normally searched on the pre-defined grid of parameter range values.

2.2 Multi-Class Classification

Binary classification problems solved by SVM have been presented; however, most of the practical problems are multi-class classification. Thus, the concept of binary classification needs to be extended to the multi-class. SVM can solve a multi-class problem by decomposing the problem to several binary problems. Three approaches, known as One-versus-All (Vapnik, 1995), One-versus-One (Debnath et al., 2004), and Directed Acyclic Graph (DAG) (Platt et al., 2000), are commonly used for the multi-class classification. These approaches can be employed by any binary classifier not restricted to SVM.

In One-versus-All, also known as One-Against-All (Cristianini and Shawe-Taylor, 2000), k binary classifiers denoted by i-vs-All are constructed, where $i = 1, \ldots, k$ and $k > 2$ for the k-class problem. Each i-vs-All changes the k-class to a two-class problem by using the training data from class i as the positive class and the data from the other classes as the negative class. For test data $\boldsymbol{x}_*$, each i-vs-All gives the corresponding decision function $f_i(\boldsymbol{x}_*)$. The data $\boldsymbol{x}_*$ belongs to class j if $f_j(\boldsymbol{x}_*) = \max_i f_i(\boldsymbol{x}_*)$ is satisfied. Figure 2.4(a) shows the diagram of One-versus-All for a four-class problem.

One-versus-One, also known as One-Against-One (Hsu and Lin, 2002), arranges the k-class problem to all possible pairs of class i and class j. Therefore it produces $\frac{k(k-1)}{2}$ binary classifiers in which data from class i are used as the positive class and data from class j are used as the negative class. To classify the class of a test data, One-versus-One uses a voting scheme. Each classifier casts a vote for the class label, which should be assigned to the test data. The test data point is then classified to the class label that receives the maximum number of votes. Figure 2.4(b) shows the One-versus-One diagram of the four-class problem.

DAG is a modification of the One-versus-One pairwise approach where the classifiers are arranged in a top-down tree structure. Each tree node represents a decision model trained by the One-versus-One method. Therefore, DAG also produces $\frac{k(k-1)}{2}$ classifiers. A test data point is classified to one leaf node label, depending on which branch of the tree is traversed. Another structure of DAG is a bottom-up tree structure (Byun and Lee, 2003) where the same $\frac{k(k-1)}{2}$ classifiers are constructed for the k-class problem, but the test data point is classified backwards from the bottom of the tree. The predicted classes from the lower levels are considered via the corresponding classifiers at the upper levels until the top of the tree is reached and a final class prediction is made at the root node. Figure (2.5) shows the structure of the top-down and bottom-up directions for DAG.

Apart from the previous approaches, in Ramanan et al. (2007)[1], we introduced a new

[1]The first two authors made equal contributions

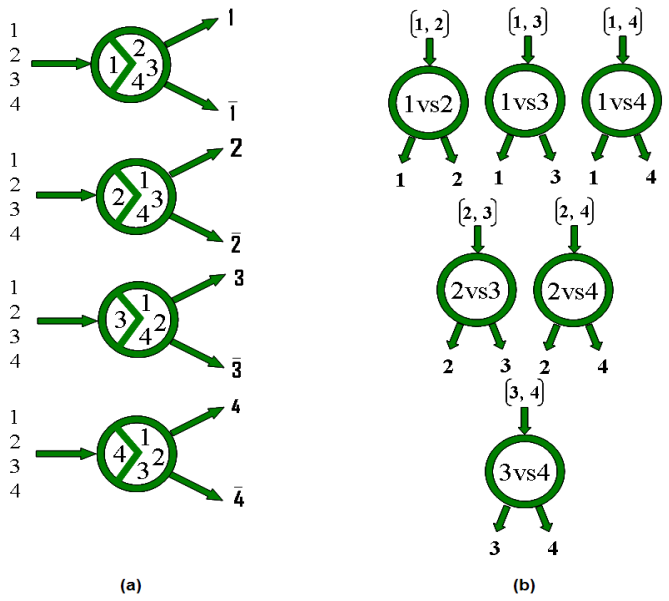


FIGURE 2.4: Classifier diagrams of (a) One-versus-All and (b) One-versus-One

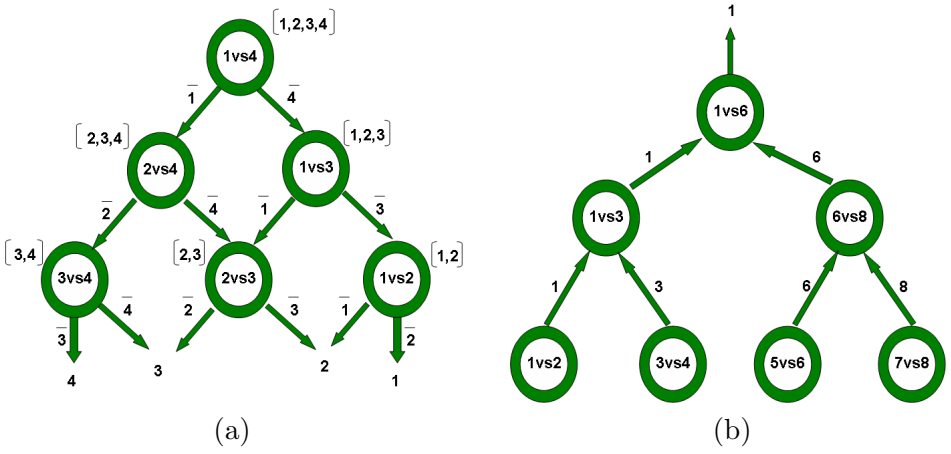


FIGURE 2.5: Two structures of DAG: (a) top-down structure (four-class problem) and (b) bottom-up structure (eight-class problem)

One-versus-All-based method called unbalanced decision tree (UDT) to deal with multi-class problems. UDT is an unbalanced directed acyclic tree structure (see Figure 2.6(b)). Any leaf node represents a class label. Each internal node is the optimal classifier chosen via training based on the One-versus-All approach, in which one selected class expressed as the positive class is evaluated against other remaining classes expressed as the negative class. During the training phase, training data used as the positive class at the upper node are removed from the training data before passing to the lower node; hence, the lowest level is pure two-class data. In each level, j-vs-All classifiers are constructed, where j is the set of classes remaining in the problem during the training phase, and the one that performs best on a subset of training data, referred to as the validation set, is chosen as the optimal classifier for the corresponding level. The pseudocode of UDT in

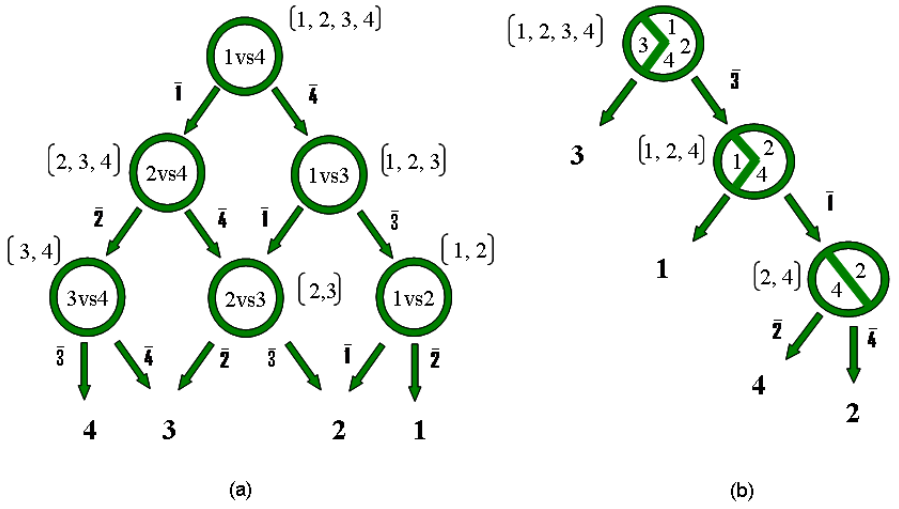


FIGURE 2.6: Comparison of classifier structure between (a) DAG and (b) UDT

the training phase is shown in Algorithm 1.

Algorithm 1 Pseudocode of UDT in the training phase

Input: training data (TR), validation data (Val), number of classes ($K > 2$), parameter ranges for tuning ($\boldsymbol{\theta}$)

Output: UDT nodes consisting of $(K-1)$ OVA classifiers and *last_class*

1: $k = \{1, \ldots, K\}$, level = 1
2: **while** $K > 2$ **do**
3: **for** each class j remaining in k **do**
4: **for** each value i in parameter ranges $\boldsymbol{\theta}$ **do** {loop for tuning parameters}
5: get accuracy A_i by running classifier j-vs-All with $\boldsymbol{\theta}_i$ on Val
6: **end for**
7: Find $\boldsymbol{\theta}_{opt}$ corresponding to maximum A_i
8: $score(j)$ = accuracy from running classifier j-vs-All with $\boldsymbol{\theta}_{opt}$ on Val
9: **end for**
10: Find j-vs-All with the maximum *score*
11: nodes(level) = j-vs-All
12: level = level+1; $k = k - \{j\}$; $K = K - 1$
13: remove data of class j from TR, Val and reset *score* to zero
14: **end while**
15: {at this step TR have only two classes; randomly pick one as j}
16: nodes(level) = j-vs-All
17: $k = k - \{j\}$; $last\_class = k$
18: construct classifier j-vs-All where its parameters are tuned as same as lines 4-7

In the prediction phase, a test data point starts from the root node and traverses the next level if the test data point is classified as the negative class, and terminates if the test data point is classified as the positive class. Therefore, UDT could give a class label at any level, depending on which level it arrives and terminates, whereas DAG has to proceed until the last level. Algorithm 2 shows the pseudocode of UDT in the

Algorithm 2 Pseudocode of UDT in the prediction phase

Input: test data ($\boldsymbol{x}_*$), number of classes (K), nodes and *last_class* from UDT training
Output: predicted class of test data

```
level = 1
while level ⩽ (K − 2) do
    model = nodes(level)
    {at this step model = j-vs-All}
    if model j-vs-All classifies x* as +1 then
        return predicted class = j
    else
        level = level+1
    end if
end while
{at this step there are only two possible predicted class left i.e. class j, k}
model = nodes(level)
if model j-vs-All classifies x* as +1 then
    return predicted class = j
else
    return predicted class = last_class
end if
```

prediction phase, while Figure 2.6 shows the structure of UDT compared to DAG for a four-class problem. During the prediction phase, UDT requires at most $k-1$ classifiers, that is, possibly only one classifier for a test data point, whereas One-versus-All requires k classifiers.

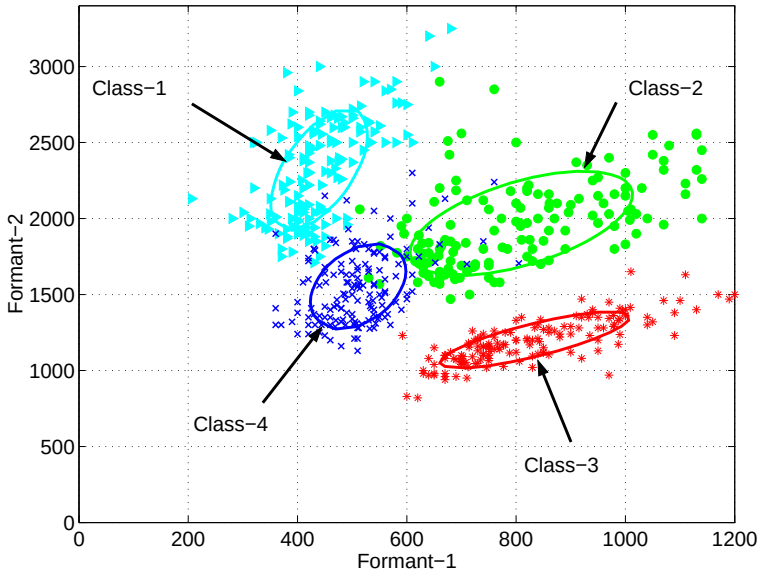


FIGURE 2.7: This figure shows the distribution of a four-class data in order to illustrate it along with the UDT construction in Figure 2.6(b). Class 3 is clearly seen as the most easily separated, followed by class 1 and 4, respectively.

Figure 2.6(b) and 2.7 illustrate the construction of UDT by using an example of the vowel dataset from Peterson and Barney (1952). As seen in Figure 2.7, class 3 is the

easiest to separate from others. UDT produces a 3-vs-All classifier at the root node. At the next lower level, training data labelled as class 3 are eliminated, as the data of class 3 are used as positive data at the previous level. The remaining data (labelled as class 1, 2 or 4) are used for training and creating the optimal classifier, that is, 1-vs-All in Figure 2.6(b). At the last level, since the data of class 1 and class 3 have already been used as positive data at the upper levels, only data labelled as class 2 or 4 are used for creating the optimal classifier, which is 4-vs-All. In the prediction step, a test data point is initially evaluated with 3-vs-All. If the test data point is predicted as positive, UDT declares that the test sample belongs to class 3. Otherwise, the test data point is then tested with the next classifier, which is 1-vs-All. Following the same rule until the last classifier 4-vs-All, the test data point is indicated by UDT as class 4 if it is positive; otherwise it is identified as class 2.

Another scheme for classifying multi-class problems is known as error correcting output code or ECOC (Dietterich and Bakiri, 1991, 1995). ECOC constructs a coding matrix $\boldsymbol{M} \in \{-1, 1\}^{K \times S}$, where K is the number of classes and S is the number of classifiers, in which each class has a unique codeword. Classifiers $f_{s=\{1,\ldots,S\}}$ are constructed in order to encode a data point as a codeword. The codeword of the test data is compared and a search is conducted for the nearest codeword in the coding matrix $\boldsymbol{M}$ by measuring either Hamming distance or loss function. The test data point is then classified to the class of the nearest codeword.

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x}) &= (f_1(\boldsymbol{x}), \cdots, f_S(\boldsymbol{x})) \\
\boldsymbol{M} &= \begin{pmatrix} m_{11} & \cdots & m_{1S} \\ \vdots & \ddots & \vdots \\ m_{K1} & \cdots & m_{KS} \end{pmatrix} \begin{matrix} class1 \\ \vdots \\ classK \end{matrix} \\
\text{class of } \boldsymbol{x} &= \underset{k}{\operatorname{argmin}}\, d(\boldsymbol{m}_k, \boldsymbol{f}(\boldsymbol{x})) \qquad (2.23) \\
\text{Hamming distance: } d(\boldsymbol{m}_k, \boldsymbol{f}(\boldsymbol{x})) &= \sum_{s=1}^{S} \frac{|m_{ks} - \text{sign}\ (f_s)|}{2} \qquad (2.24) \\
\text{Loss function: } d(\boldsymbol{m}_k, \boldsymbol{f}(\boldsymbol{x})) &= \sum_{s=1}^{S} L(m_{ks} f_s) \qquad (2.25) \\
\text{e.g. linear loss } L(m_{ks} f_s) &= -m_{ks} f_s
\end{aligned}
$$

Allwein et al. (2000) define the element in the coding matrix $\boldsymbol{M}$ as $m_{ks} \in \{-1, 0, 1\}, k = 1, \ldots, K$ and $s = 1, \ldots, S$, where $m_{ks} = 1(-1)$ when the data in class k are used for positive (negative) class and $m_{ks} = 0$ when data in class k are not used in classifier $f_s(\boldsymbol{x})$. Hence, bit 0 is treated as a don't-care bit and $\boldsymbol{m}_k$ denotes the kth row of $\boldsymbol{M}$ representing the codeword of class k. The following matrices show two examples of One-versus-All (OVA) and One-versus-One (OVO) in the scheme of ECOC on a three-class

problem.

$$\boldsymbol{M}_{OVA} = \begin{pmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}$$

$$\boldsymbol{M}_{OVO} = \begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{pmatrix}$$

For the $\boldsymbol{M}_{OVA}$ example, the column classifiers $\boldsymbol{f}(\boldsymbol{x})$ are 1-vs-All, 2-vs-All and 3-vs-All. Hence, the codeword of class 1 is (1 -1 -1); consequently, the data of class 1 are used as positive class for the first column classifier, whereas they are used as negative class data for the other two classifiers. For the $\boldsymbol{M}_{OVO}$ example, the column classifiers $\boldsymbol{f}(\boldsymbol{x})$ are 1-vs-2, 1-vs-3 and 2-vs-3. Hence, the codeword of class 2 is (-1 0 1). Consequently, the data of class 2 are used as positive class for the last column classifier but negative class for the first classifier and are not used for classifier 1-vs-3. The design of the coding matrix plays a significant role in ECOC's performance. Dietterich and Bakiri (1995) suggested the properties of the coding matrix, but it is still an open question for a large number of classes for constructing an efficient coding matrix.

2.3 Gaussian Process Classification

Gaussian processes have recently been very attractive to machine learning community not only as a model for non-parametric regression problems for which it is naturally suited, but also to solve classification problems. While SVM is a deterministic approach generating a function that predicts class labels, Gaussian processes have the Bayesian probabilistic appeal, which is the ability to summarise uncertainties associated with the inference process as predictive probability densities in addition to the class prediction. In this section, the concept of Gaussian processes for classification problems is explained. Recall that a data point is represented by a vector $\boldsymbol{x} = (x_1,\ x_2,\ \ldots,\ x_d)^T$ where d is the number of features and the associated class label is denoted by $y \in \{-1, +1\}$. Therefore, a training dataset consists of $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1, \ldots, N\}$ where N is the number of training data. A model or classifier based on this training data is constructed and used for predicting the class label (y_*) of a new unseen test data point ($\boldsymbol{x}_*$). Bayes' theorem states that the posterior probability distribution is obtained by the multiplication between prior probability and the likelihood function and then normalising by marginal likelihood. It can be written as:

$$p(y|\boldsymbol{x}) = \frac{p(y)\ p(\boldsymbol{x}|y)}{p(\boldsymbol{x})}. \tag{2.26}$$

The predictive probability for positive class of the test data generated by GP reflects the level of confidence of the classifier concerning whether the data should be classified as positive class. If the predictive probability is low, it implies that the test data should be classified as negative class. The predictive value 0.5 is generally set as the threshold to make a class prediction for the given test data.

Rasmussen and Williams (2006) define a Gaussian process as a stochastic process that assumes a Gaussian joint distribution of latent function values of the associated input $\boldsymbol{x}$ and place as the prior of the latent function. The prior is distributed according to:

$$\boldsymbol{f}|\boldsymbol{X},\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{f}|\boldsymbol{0},\boldsymbol{C}) \tag{2.27}$$

where $\boldsymbol{f} = [f(\boldsymbol{x}_1),\ f(\boldsymbol{x}_2),\ \ldots, f(\boldsymbol{x}_N)\ ]^T$ is the vector of latent function values corresponding to the input, $\boldsymbol{X} = [\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ \ldots,\ \boldsymbol{x}_N]^T$ and $\boldsymbol{C}$ is the positive definite covariance matrix governed by hyperparameters $\boldsymbol{\theta}$. The covariance function plays an important role in GP as it encodes our assumptions about the function we want to learn. In this thesis, the squared exponential covariance function $\boldsymbol{C}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \sigma_f^2 \exp\left(-|\boldsymbol{x}_i - \boldsymbol{x}_j|^2/2l^2\right)$ is used for GP, in which there are two parameters, namely the signal variance σ_f^2 and the length-scale l. We select the squared exponential covariance function as it is a function of $|\boldsymbol{x}_i - \boldsymbol{x}_j|$, which is invariant to translation in the input space, and its form is similar to the RBF kernel in SVM.

GP first determines the distribution of the latent function corresponding to the test data $p(f_*|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y})$ and later uses it to calculate the predictive probability of the test data $p(y_* = 1|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y})$. These are operated via:

$$\begin{aligned} p(f_*|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y}) &= \int p(f_*,\boldsymbol{f}|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y})\,d\boldsymbol{f} \\ &= \int p(f_*|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{f})\,p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})\,d\boldsymbol{f} \qquad (2.28) \\ p(y_* = 1|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y}) &= \int \tau(f_*)\,p(f_*|\boldsymbol{x}_*,\boldsymbol{X},\boldsymbol{y})\,df_* \qquad (2.29) \end{aligned}$$

where $p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})$ is the posterior of the latent function which is calculated by Bayes' theorem: $p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y}) = p(\boldsymbol{f}|\boldsymbol{X})\,p(\boldsymbol{y}|\boldsymbol{f})/p(\boldsymbol{y}|\boldsymbol{X})$. Given the latent function, the joint likelihood of class labels is:

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{N} p(y_i|f(\boldsymbol{x}_i)) = \prod_{i=1}^{N} \tau(y_i f(\boldsymbol{x}_i)) \tag{2.30}$$

where τ is either logistic (σ) or a cumulative density function of the standard normal distribution (Φ):

$$\sigma(y_i f(\boldsymbol{x}_i)) = \frac{1}{1+e^{-(y_i f(\boldsymbol{x}_i))}} \tag{2.31}$$

$$\begin{aligned}\Phi(y_i f(\boldsymbol{x}_i)) &= \int_{-\infty}^{y_i f(\boldsymbol{x}_i)} \mathcal{N}(z|0,1)\, dz \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{y_i f(\boldsymbol{x}_i)} \exp\left(-\frac{z^2}{2}\right) dz.\end{aligned} \tag{2.32}$$

The likelihood τ has the symmetric property, that is, $\tau(-z) = 1 - \tau(z)$. Therefore, the posterior over the latent function can be written as the multiplication between Gaussian prior and the joint likelihood divided by marginal likelihood (Kuss and Rasmussen, 2006):

$$p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) = \frac{\mathcal{N}(\boldsymbol{f}|\boldsymbol{0},\boldsymbol{C})}{p(\mathcal{D}|\boldsymbol{\theta})} \prod_{i=1}^{N} \tau(y_i f(\boldsymbol{x}_i)) \tag{2.33}$$

where the marginal likelihood is $p(\mathcal{D}|\boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{\theta})\ d\boldsymbol{f}$. Note that $\mathcal{D}$ is $\{(\boldsymbol{X},\boldsymbol{y})\}$ and $\boldsymbol{\theta}$ is the hyperparameters. The posterior, however, cannot be solved analytically, as the joint likelihood in equation (2.30) is not Gaussian. Hence, an approximation is required to solve this problem. A number of the approximation, Laplace approximation (Williams and Barber, 1998), expectation propagation (Minka, 2001) and variational approximation (Gibbs and MacKay, 2000), are described in the following subsections.

2.3.1 Laplace approximation

The Laplace approximation method approximates the posterior $p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta})$ by taking the Taylor expansion of the logarithm of the posterior up to the second order term around the maximum of the posterior. The marginal likelihood is ignored here because it is independent of $\boldsymbol{f}$ when maximising with respect to $\boldsymbol{f}$; hence, the method only considers the numerator of equation (2.33). The logarithm of unnormalised posterior is the sum of the logarithm of the likelihood and the logarithm of the Gaussian prior. The logarithm of the posterior is shown as follows:

$$\log \mathcal{N}(\boldsymbol{f}|\boldsymbol{0},\boldsymbol{C}) = -\tfrac{1}{2}\boldsymbol{f}^T\boldsymbol{C}^{-1}\boldsymbol{f} - \tfrac{1}{2}\log|\boldsymbol{C}| - \tfrac{d}{2}\log 2\pi \tag{2.34}$$

$$\begin{aligned}\log p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) &\rightarrow \log \mathcal{N}(\boldsymbol{f}|\boldsymbol{0},\boldsymbol{C}) + \log p(\boldsymbol{y}|\boldsymbol{f}) \\ &= \log p(\boldsymbol{y}|\boldsymbol{f}) - \tfrac{1}{2}\boldsymbol{f}^T\boldsymbol{C}^{-1}\boldsymbol{f} - \tfrac{1}{2}\log|\boldsymbol{C}| - \tfrac{d}{2}\log 2\pi.\end{aligned} \tag{2.35}$$

The Gaussian approximation of the above logarithm of posterior distribution shown in Rasmussen and Williams (2006) is in the form of:

$$q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}},\boldsymbol{A}^{-1}) \propto \exp\left(-\tfrac{1}{2}(\boldsymbol{f}-\hat{\boldsymbol{f}})^T\,\boldsymbol{A}\,(\boldsymbol{f}-\hat{\boldsymbol{f}})\right) \tag{2.36}$$

where $\hat{\boldsymbol{f}}$ is the maximum of the posterior and $\boldsymbol{A} = -\nabla\nabla \log\, p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta})|_{\boldsymbol{f}=\hat{\boldsymbol{f}}}$ is the Hessian of the negative logarithm of the posterior at the point yielding the maximum posterior. The maximum posterior point and covariance matrix $\boldsymbol{A}^{-1}$ can be found by the first and second derivatives of equation (2.35), respectively, that is:

$$\begin{aligned} \nabla \log\, p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) &= \nabla \log\, p(\boldsymbol{y}|\boldsymbol{f}) - \boldsymbol{C}^{-1}\boldsymbol{f} \\ \hat{\boldsymbol{f}} &= \boldsymbol{C}(\nabla\, \log\, p(\boldsymbol{y}|\hat{\boldsymbol{f}})) \qquad (2.37) \\ \nabla\nabla\, \log\, p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) &= \nabla\nabla\, \log\, p(\boldsymbol{y}|\boldsymbol{f}) - \boldsymbol{C}^{-1} \\ &= -\boldsymbol{W} - \boldsymbol{C}^{-1} \qquad (2.38) \end{aligned}$$

where $\boldsymbol{W}$ is $-\nabla\nabla\, \log\, p(\boldsymbol{y}|\boldsymbol{f})$. Therefore, the posterior from equation (2.33) is approximated by:

$$q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{f}|\hat{\boldsymbol{f}},(\boldsymbol{W}+\boldsymbol{C}^{-1})^{-1}) \tag{2.39}$$

Rasmussen and Williams (2006) point out that equation (2.37) cannot be solved directly, and they use Newton's method to find $\hat{\boldsymbol{f}}$ with the iterative equation (2.40) until convergence:

$$\begin{aligned} \boldsymbol{f}^{new} &= \boldsymbol{f} - (\nabla\nabla\, \log\, p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}))^{-1}\, \nabla\, \log\, p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) \\ &= \boldsymbol{f} + (\boldsymbol{W}+\boldsymbol{C}^{-1})^{-1}\,(\nabla\, \log\, p(\boldsymbol{y}|\boldsymbol{f}) - \boldsymbol{C}^{-1}\boldsymbol{f}) \\ &= (\boldsymbol{W}+\boldsymbol{C}^{-1})^{-1}\,(\boldsymbol{W}\boldsymbol{f} + \nabla\, \log\, p(\boldsymbol{y}|\boldsymbol{f})). \qquad (2.40) \end{aligned}$$

Subsequently, the approximate posterior (2.39) is substituted in (2.28) obtaining (2.41) and the predictive probability of positive class (2.29) becomes (2.42), respectively

$$\begin{aligned} p(f_*|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta}) &\approx \int p(f_*|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta},\boldsymbol{f})\, q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta})\, d\boldsymbol{f} \\ &= q(f_*|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta}) \qquad (2.41) \\ p(y_*=1|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta}) &\approx \int \tau(f_*)\, q(f_*|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta})\, df_* \\ &= q(y_*=1|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta}) \qquad (2.42) \end{aligned}$$

where $q(f_*|\boldsymbol{x}_*,\mathcal{D},\boldsymbol{\theta})$ is Gaussian with mean $\boldsymbol{c}(\boldsymbol{x}_*)^T\, \nabla\, \log\, p(\boldsymbol{y}|\hat{\boldsymbol{f}})$ and variance $c(\boldsymbol{x}_*,\boldsymbol{x}_*) - \boldsymbol{c}(\boldsymbol{x}_*)^T(\boldsymbol{C}+\boldsymbol{W}^{-1})^{-1}\boldsymbol{c}(\boldsymbol{x}_*)$. $\boldsymbol{c}(\boldsymbol{x}_*)$ is the vector of covariances between the latent function of the test data and all N training data and $c(\boldsymbol{x}_*,\boldsymbol{x}_*)$ is the covariance between $f(\boldsymbol{x}_*)$ and $f(\boldsymbol{x}_*)$.

2.3.2 Expectation propagation

The idea behind the expectation propagation (EP) algorithm is to approximate the intractable interested distribution by a tractable class of distribution from an exponential family such as Gaussian distribution. The exponential family of a distribution over $\boldsymbol{x}$ given parameters $\boldsymbol{\theta}$ can be written in a form of $p(\boldsymbol{x}|\boldsymbol{\theta}) = B(\boldsymbol{\theta})h(\boldsymbol{x})\exp\{A(\boldsymbol{\theta})u(\boldsymbol{x})\}$. In Gaussian process classification, EP approximates the joint likelihood term by unnormalised Gaussian function in the latent function:

$$\begin{aligned}\prod_{i=1}^{N} p(y_i|f(\boldsymbol{x}_i)) \quad &\approx \quad \prod_{i=1}^{N}\Psi_i \quad = \quad \prod_{i=1}^{N}\Psi_i(f(\boldsymbol{x}_i)|\tilde{Z}_i,\tilde{\mu}_i,\tilde{\sigma}_i^2)\\ &= \quad \mathcal{N}(\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\Sigma}})\prod_i \tilde{Z}_i\ ,\end{aligned}$$

$$\tilde{\boldsymbol{\Sigma}} = \begin{pmatrix} \tilde{\sigma}_1^2 & 0 & 0 & \cdots & 0 & 0\\ 0 & \tilde{\sigma}_2^2 & 0 & \cdots & 0 & 0\\ 0 & 0 & \ddots & \cdots & 0 & 0\\ 0 & 0 & 0 & \tilde{\sigma}_i^2 & 0 & 0\\ 0 & 0 & 0 & 0 & \ddots & 0\\ 0 & 0 & 0 & 0 & 0 & \tilde{\sigma}_N^2 \end{pmatrix}, \quad \tilde{\boldsymbol{\mu}} = \begin{pmatrix} \tilde{\mu}_1\\ \tilde{\mu}_2\\ \vdots\\ \tilde{\mu}_N \end{pmatrix}. \tag{2.43}$$

Therefore, the posterior equation (2.33) changes into:

$$\begin{aligned} p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) \quad \approx \quad q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) \quad &\propto \quad p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{\theta})\prod_{i=1}^{N}\Psi_i\\ &= \quad \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})\ ,\\ \boldsymbol{\mu} \ = \ \boldsymbol{\Sigma}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{\mu}}, \quad \boldsymbol{\Sigma} \quad &= \quad (\boldsymbol{C}^{-1}+\tilde{\boldsymbol{\Sigma}}^{-1})^{-1}. \end{aligned} \tag{2.44}$$

To obtain the appropriate $q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta})$, EP iteratively seeks Ψ_i that minimise the following Kullback-Leibler divergence (KL) where the hyperparameters $\boldsymbol{\theta}$ are implicitly embedded in the latent function f, and subscript i denotes the corresponding case of data $\boldsymbol{x}_i$ for concise notations:

$$\begin{aligned}\Psi_i^{new} \quad &= \quad \underset{\Psi_i}{\operatorname{argmin}}\, KL\left(\frac{q(\boldsymbol{f}|\mathcal{D})}{\Psi_i^{old}}\,p(y_i|f_i)\ \middle\|\ \frac{q(\boldsymbol{f}|\mathcal{D})}{\Psi_i^{old}}\Psi_i\right)\\ &= \quad \underset{\Psi_i}{\operatorname{argmin}}\, KL\left(q_{-i}(f_i|\mathcal{D})\,p(y_i|f_i)\ \middle\|\ q_{-i}(f_i|\mathcal{D})\,\Psi_i\right)\end{aligned} \tag{2.45}$$

where the subscript $-i$ indicates the case that the data $\boldsymbol{x}_i$ is not included, and the Kullback-Leibler is defined by $KL(p(x)\|q(x)) = \int p(x)\log\frac{p(x)}{q(x)}\,dx$. The parameters needed for EP approximation and their full derivations are shown in Rasmussen and

Williams (2006, Chapter 3), and they are summarised by the following equations:

$$\begin{aligned}
\mu_{-i} &= \sigma_{-i}^2(\sigma_i^{-2}\mu_i - \tilde{\sigma}_i^{-2}\tilde{\mu}_i) \quad , \sigma_{-i}^2 = (\sigma_i^{-2} - \tilde{\sigma}_i^{-2})^{-1} \quad , \sigma_i^2 = \Sigma_{ii} \\
\hat{Z}_i &= \tau(z_i), \qquad \hat{\mu}_i = \mu_{-i} + \frac{y_i\sigma_{-i}^2\mathcal{N}(z_i)}{\tau(z_i)\sqrt{1+\sigma_{-i}^2}}, \quad z_i = \frac{y_i\mu_{-i}}{\sqrt{1+\sigma_{-i}^2}} \\
\hat{\sigma}_i^2 &= \sigma_{-i}^2 - \frac{\sigma_{-i}^4\mathcal{N}(z_i)}{(1+\sigma_{-i}^2)\tau(z_i)}\left(z_i + \frac{\mathcal{N}(z_i)}{\tau(z_i)}\right) \qquad (2.46) \\
\tilde{\mu}_i &= \tilde{\sigma}_i^2(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_{-i}^{-2}\mu_{-i}), \qquad \tilde{\sigma}_i^2 = (\hat{\sigma}_i^{-2} - \sigma_{-i}^{-2})^{-1}, \qquad (2.47) \\
\tilde{Z}_i &= \hat{Z}_i\sqrt{2\pi}\sqrt{\sigma_{-i}^2 + \tilde{\sigma}_i^2}\,\exp\left(\tfrac{1}{2}(\mu_{-i} - \tilde{\mu}_i)^2/(\sigma_{-i}^2 + \tilde{\sigma}_i^2)\right). \qquad (2.48)
\end{aligned}$$

Once a local cite Ψ_i is updated, the approximate posterior is updated by equation (2.44) and EP iteratively updates Ψ_i until convergence. The prediction of the unseen test data by EP also uses the similar equations as in the Laplace approximation equation (2.41) and (2.42) whereas $q(f_*|\boldsymbol{x}_*, \mathcal{D}, \boldsymbol{\theta})$ by EP is Gaussian with the mean $\boldsymbol{c}(\boldsymbol{x}_*)^T(\boldsymbol{C} + \tilde{\boldsymbol{\Sigma}})^{-1}\tilde{\boldsymbol{\mu}}$ and variance $c(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{c}(\boldsymbol{x}_*)^T(\boldsymbol{C} + \tilde{\boldsymbol{\Sigma}})^{-1}\boldsymbol{c}(\boldsymbol{x}_*)$.

2.3.3 Variational approximation

Variational approximation (VA) approximates the posterior distribution by bounding on each likelihood term. Gibbs and MacKay (2000) define the lower bound on each logistic likelihood function as the following:

$$\begin{aligned}
p(y_i|f_i) &\geqslant g(\nu_i)\,\exp\left(\tfrac{1}{2}(f_i - \nu_i) - \lambda_i(f_i^2 - \nu_i^2)\right) \\
&= q(y_i|f_i, \nu_i) \qquad (2.49)
\end{aligned}$$

where ν_i is a variational parameter for each data point $\boldsymbol{x}_i$, $g(\nu_i) = \dfrac{1}{1+\exp(-\nu_i)}$ and $\lambda_i = \lambda(\nu_i) = \dfrac{\left(g(\nu_i) - \frac{1}{2}\right)}{2\nu_i}$. Substituting the lower bound, the likelihood term then changes to:

$$\begin{aligned}
p(\boldsymbol{y}|\boldsymbol{f}) &\geqslant \prod_{i=1}^{N} q(y_i|f_i, \nu_i) \\
&\propto \exp(\boldsymbol{f}^T\boldsymbol{\Lambda}\boldsymbol{f} + (\boldsymbol{b}\odot\boldsymbol{y})^T\boldsymbol{f} + \boldsymbol{e}^T\mathbf{1}) \qquad (2.50)
\end{aligned}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix whose diagonal elements are $-\lambda_i$. The Newton's method is applied for determining these parameters, which are tight at $f_i = \pm\,\nu_i$ (Nickisch and Rasmussen, 2008). The coefficient $\boldsymbol{b}$ is a column vector of which all elements are $\frac{1}{2}$. The operator $\odot$ is the element-wise product of two vectors and $\mathbf{1}$ denotes the column vector with all elements of 1. The coefficients e_i in $\boldsymbol{e}$ are $\nu_i^2\lambda_i - \frac{1}{2}\nu_i + \ln g(\nu_i)$. Similar to equation (2.33), the posterior over latent function approximated by VA is subsequently

obtained by multiplying the lower bound of the likelihood to the Gaussian prior:

$$\begin{aligned} p(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) \quad &\approx \quad q(\boldsymbol{f}|\mathcal{D},\boldsymbol{\theta}) \quad \propto \quad p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{\theta}) \prod_{i=1}^{N} q(y_i|f_i,\nu_i) \\ &= \quad \mathcal{N}\left(\boldsymbol{\mu},(\boldsymbol{W}+\boldsymbol{C}^{-1})^{-1}\right) , \\ \boldsymbol{W} \quad &= \quad -2\boldsymbol{\Lambda}, \quad \boldsymbol{\mu} \quad = \quad (\boldsymbol{W}+\boldsymbol{C}^{-1})^{-1}(\boldsymbol{y}\odot\boldsymbol{b}). \end{aligned} \tag{2.51}$$

For the test data prediction, the expectation of the posterior is approximated by the Gaussian with the mean $\boldsymbol{c}(\boldsymbol{x}_*)^T(\boldsymbol{I}+2\boldsymbol{\Lambda}\boldsymbol{C})^{-1}(\boldsymbol{b}\odot\boldsymbol{y})$ and variance $c(\boldsymbol{x_*},\boldsymbol{x_*})+2\boldsymbol{c}(\boldsymbol{x}_*)^T(\boldsymbol{I}+2\boldsymbol{\Lambda}\boldsymbol{C})^{-1}\boldsymbol{\Lambda}\boldsymbol{c}(\boldsymbol{x}_*)$, where $\boldsymbol{I}$ is the NxN identity matrix (Gibbs and MacKay, 2000). Nickisch and Rasmussen (2008) also show the lower bounding likelihood when the cumulative Gaussian likelihood, known as probit function, is used. The probit likelihood bound is:

$$p(y_i|f_i) \geqslant \exp(a_i f_i^2 + b_i y_i f_i + e_i) \tag{2.52}$$

where $a_i = -\frac{1}{2}, b_i = \nu_i + \dfrac{\mathcal{N}(\nu_i)}{\Phi(\nu_i)}$ and $e_i = (\frac{\nu_i}{2} - b_i)\nu_i + \ln(\Phi(\nu_i))$.

2.3.4 Sparse Gaussian

A particular difficulty in the use of Gaussian processes is scalability. The approach requires the inversion of a matrix whose number of dimensions is the same as the number of data points in the problem. Sparse approximations to Gaussian processes have been motivated mainly by this scalability issue. Informative vector machine or IVM proposed by Lawrence et al. (2003) is a fast sparse Gaussian approximation in which a subset of training data, known as an active set, is selected with criteria based on the differential entropy score. The data which have a large reduction in the entropy are selected for the active set. The concept is similar to the expectation propagation in the sense that IVM approximates the true posterior distribution , $p(\boldsymbol{f}|\mathcal{D})$, by a Gaussian $q(\boldsymbol{f}|\mathcal{D})$, which updates the likelihood of the local sites. However, IVM updates for the active set data while the local site parameters of the non-active set are set to zeros. The Gaussian approximation is:

$$\begin{aligned} q(\boldsymbol{f}|\mathcal{D}) \quad &\propto \quad p(\boldsymbol{f}|\boldsymbol{X}) \prod_{i=1}^{N} \exp\left(-\frac{p_i}{2}(f_i - m_i)^2\right) \\ &= \quad \mathcal{N}(\boldsymbol{f}|\boldsymbol{h},\boldsymbol{A}) \\ \boldsymbol{h} \quad &= \quad \boldsymbol{A}\boldsymbol{W}\boldsymbol{m} \\ \boldsymbol{A} \quad &= \quad (\boldsymbol{C}^{-1}+\boldsymbol{W})^{-1} = \boldsymbol{C} - \boldsymbol{M}^T\boldsymbol{M} \\ \boldsymbol{W} \quad &= \quad \mathrm{diag}(\boldsymbol{p}) \\ \boldsymbol{M} \quad &= \quad \boldsymbol{L}^{-1}\boldsymbol{W}_I^{\frac{1}{2}}\boldsymbol{C}_{I,\cdot} \end{aligned} \tag{2.53}$$

where $\boldsymbol{C}$ is the covariance matrix of the Gaussian prior, $p(\boldsymbol{f}|\boldsymbol{X})$, and $\boldsymbol{L}$ is the lower triangular of Cholesky decomposition $\boldsymbol{B} = \boldsymbol{I} + \boldsymbol{W}_I^{\frac{1}{2}} \boldsymbol{C}_I \boldsymbol{W}_I^{\frac{1}{2}} = \boldsymbol{L}\boldsymbol{L}^T$. I denotes the active set and the components of $\boldsymbol{p}$ and $\boldsymbol{m}$ not in I are set to zeros. $\boldsymbol{C}_{I,\cdot}$ is a submatrix of $\boldsymbol{C}$ whose rows are indicated by the active set. The following equations are used for updating the ith site parameters when the ith data are selected to the active set:

$$\begin{aligned} p_i &= \frac{\nu_i}{1 - a_{i,i}\nu_i} && (2.54) \\ m_i &= h_i + \frac{\alpha_i}{\nu_i} && (2.55) \\ \alpha_i &= \frac{y_i \cdot \mathcal{N}(z_i|0,1)}{\Phi(z_i)\sqrt{1 + a_{i,i}}} \\ z_i &= \frac{y_i \cdot (h_i + b)}{\sqrt{1 + a_{i,i}}} \\ \nu_i &= \alpha_i \left(\alpha_i + \frac{h_i + b}{1 + a_{i,i}} \right) \end{aligned}$$

Note that the bias ,b, appears in the likelihood term $p(y_i|f_i) = \Phi(y_i(f_i + b))$. Lawrence et al. (2003) append the row $(\boldsymbol{l}^T, l)$ and $\boldsymbol{\mu}^T$ to $\boldsymbol{L}$ and $\boldsymbol{M}$, respectively for updating the corresponding matrices, where:

$$\boldsymbol{l} = \sqrt{p_i}\boldsymbol{M}_{\cdot,i}, \quad l = \sqrt{1 + p_i \boldsymbol{C}_{i,i} - \boldsymbol{l}^T\boldsymbol{l}}, \quad \boldsymbol{\mu} = l^{-1}(\sqrt{p_i}\boldsymbol{C}_{\cdot,i} - \boldsymbol{M}^T\boldsymbol{l}). \tag{2.56}$$

The subscript $\cdot, i$ means all components in column i, and then $\boldsymbol{h}$ and $\boldsymbol{A}$ are updated by $\boldsymbol{h} = \boldsymbol{h} + \alpha_i l p_i^{-\frac{1}{2}} \boldsymbol{\mu}$, and $\mathrm{diag}(\boldsymbol{A}) = \mathrm{diag}(\boldsymbol{A}) - (\mu_j^2)_j$, respectively. The differential entropy score of the jth $\in J$, where $I \cup J = \mathcal{D}, I \cap J = \{\}$, is calculated by:

$$\Delta H_j = \frac{1}{2}\log(1 - a_{j,j}\nu_j) \tag{2.57}$$

and then the data j which has maximum ΔH_j is included in the active set I and removed from J.

Another sparse Gaussian process developed by Naish-Guzman and Holden (2008) known as "fully independent training conditional (FITC) approximation" in which pseudo-inputs or inducing inputs are termed. The inducing inputs $\bar{\boldsymbol{X}} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M]^T$, where M is the number of inducing inputs, are associated with latent values $\boldsymbol{u}$. FITC approximates the joint prior over training and test data by:

$$\begin{aligned} p\left(\boldsymbol{f}, \boldsymbol{f}_*|\boldsymbol{X}, \boldsymbol{X}_*, \bar{\boldsymbol{X}}\right) &= \int p\left(\boldsymbol{f}, \boldsymbol{f}_*|\boldsymbol{u}, \boldsymbol{X}, \boldsymbol{X}_*\right) p(\boldsymbol{u}|\bar{\boldsymbol{X}})\, d\boldsymbol{u} \\ &\approx \int q(\boldsymbol{f}|\boldsymbol{u}, \boldsymbol{X})\, q(\boldsymbol{f}_*|\boldsymbol{u}, \boldsymbol{X}_*)\, p\left(\boldsymbol{u}|\bar{\boldsymbol{X}}\right) d\boldsymbol{u} \end{aligned} \tag{2.58}$$

where each term is approximated by the following Gaussians:

$$
\begin{aligned}
q(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{X}) &= \mathcal{N}\left(\boldsymbol{f};\ \boldsymbol{C_{fu}}\boldsymbol{C_{uu}^{-1}}\boldsymbol{u},\ \mathrm{diag}(\boldsymbol{C_{ff}}-\boldsymbol{Q_{ff}})\right) && (2.59)\\
q(\boldsymbol{f}_*|\boldsymbol{u},\boldsymbol{X}_*) &= \mathcal{N}\left(\boldsymbol{f}_*;\ \boldsymbol{C_{*u}}\boldsymbol{C_{uu}^{-1}}\boldsymbol{u},\ \mathrm{diag}(\boldsymbol{C_{**}}-\boldsymbol{Q_{**}})\right) && (2.60)\\
p\left(\boldsymbol{u}|\bar{\boldsymbol{X}}\right) &= \mathcal{N}(\boldsymbol{u};\boldsymbol{0},\boldsymbol{C_{uu}}) && (2.61)
\end{aligned}
$$

where $\boldsymbol{Q_{ab}} = \boldsymbol{C_{au}}\boldsymbol{C_{uu}^{-1}}\boldsymbol{C_{ub}}$. The prior on $\boldsymbol{f}$ can be approximated by using equation (2.59) and (2.61):

$$
\begin{aligned}
q(\boldsymbol{f}|\boldsymbol{X}) &= \int q(\boldsymbol{f}|\boldsymbol{u},\boldsymbol{X})\, p\left(\boldsymbol{u}|\bar{\boldsymbol{X}}\right)\, d\boldsymbol{u}\\
&= \mathcal{N}\left(\boldsymbol{f};\ \boldsymbol{0},\ \boldsymbol{Q_{ff}} + \mathrm{diag}(\boldsymbol{C_{ff}}-\boldsymbol{Q_{ff}})\right). && (2.62)
\end{aligned}
$$

Naish-Guzman and Holden (2008) use EP with the above prior and probit likelihood function to come up with the posterior approximation $q(\boldsymbol{f}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{f};\boldsymbol{h},\boldsymbol{A})$. They then find the posterior $p(\boldsymbol{u}|\boldsymbol{y})$ by using the following:

$$
\begin{aligned}
p(\boldsymbol{u}|\boldsymbol{f}) &\propto p(\boldsymbol{f}|\boldsymbol{u})\, p(\boldsymbol{u})\\
&= \mathcal{N}(\boldsymbol{u};\ \boldsymbol{R_0^{-1}}\boldsymbol{s},\boldsymbol{R_0^{-1}}\boldsymbol{S}\boldsymbol{R_0^{-T}}) && (2.63)\\
p(\boldsymbol{u}|\boldsymbol{y}) &\approx \int p(\boldsymbol{u}|\boldsymbol{f})\, q(\boldsymbol{f}|\boldsymbol{y})\, d\boldsymbol{f}\\
&= \mathcal{N}(\boldsymbol{u};\ \boldsymbol{R_0^{-1}}\boldsymbol{\mu},\boldsymbol{R_0^{-1}}\boldsymbol{\Sigma}\boldsymbol{R_0^{-T}}) && (2.64)
\end{aligned}
$$

where $\boldsymbol{\mu} = \boldsymbol{S}\boldsymbol{R}_0\boldsymbol{P}_0^T\boldsymbol{D}_0^{-1}\boldsymbol{h},\ \boldsymbol{\Sigma} = \boldsymbol{S}+\boldsymbol{S}\boldsymbol{R}_0\boldsymbol{P}_0^T\boldsymbol{D}_0^{-1}\boldsymbol{A}\boldsymbol{D}_0^{-1}\boldsymbol{P}_0\boldsymbol{R}_0^T\boldsymbol{S},\ \boldsymbol{s} = \boldsymbol{S}\boldsymbol{R}_0\boldsymbol{P}_0^T\boldsymbol{D}_0^{-1}\boldsymbol{f},\ \boldsymbol{S}^{-1} = \boldsymbol{I} + \boldsymbol{R}_0\boldsymbol{P}_0^T\boldsymbol{D}_0^{-1}\boldsymbol{P}_0\boldsymbol{R}_0^T$. The zero subscript denotes the initial value that will be updated through the EP iterations. $\boldsymbol{D}$ is the $\mathrm{diag}(\boldsymbol{C_{ff}}-\boldsymbol{Q_{ff}})$, $\boldsymbol{P}_0 = \boldsymbol{C_{fu}}$ and $\boldsymbol{R}$ is the upper triangular Cholesky factor of $\boldsymbol{M} = \boldsymbol{R}^T\boldsymbol{R}$ where $\boldsymbol{M}_0 = \boldsymbol{C_{uu}^{-1}}$. To make a prediction on a test data point, equation (2.60) is integrated out and passed to the probit function as follows:

$$
\begin{aligned}
p(f_*|\boldsymbol{x}_*,\boldsymbol{y}) &= \int p(f_*|\boldsymbol{u})\, p(\boldsymbol{u}|\boldsymbol{y})\, d\boldsymbol{u}\\
&= \mathcal{N}(f_*;\ \mu_*,\sigma_*^2) && (2.65)\\
p(y_*|\boldsymbol{x}_*,\boldsymbol{y}) &= \int p(y_*|f_*)\, p(f_*|\boldsymbol{x}_*,\boldsymbol{y})\, df_*\\
&= \Phi\left(\frac{y_*\mu_*}{\sqrt{1+\sigma_*^2}}\right) && (2.66)
\end{aligned}
$$

where $\mu_* = \boldsymbol{c}_*^T\boldsymbol{R}_0^T\mu$ and $\sigma_* = \boldsymbol{c}_{**} + \boldsymbol{c}_*^T\boldsymbol{R}_0^T(\boldsymbol{\Sigma}-\boldsymbol{I})\boldsymbol{R}_0\boldsymbol{c}_*$.

2.4 Fisher's Linear Discriminant

Fisher's linear discriminant is a dimensionality reduction method that projects high dimensional data onto one dimensional line. The two-class data projection should be as well separated as possible by maximising the generalized Fisher's criterion ($J(\boldsymbol{w})$):

$$
\begin{aligned}
J(\boldsymbol{w}) &= \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}} && (2.67)\\
\boldsymbol{S}_B &= (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \\
\boldsymbol{S}_W &= \sum_{i \in c_1} (\boldsymbol{x}_i - \boldsymbol{\mu}_1)(\boldsymbol{x}_i - \boldsymbol{\mu}_1)^T + \sum_{i \in c_2} (\boldsymbol{x}_i - \boldsymbol{\mu}_2)(\boldsymbol{x}_i - \boldsymbol{\mu}_2)^T \\
\boldsymbol{w} &\propto \boldsymbol{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) && (2.68)
\end{aligned}
$$

where $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are the mean of positive and negative class data, respectively. $\boldsymbol{S}_B$ and $\boldsymbol{S}_W$ are the between-class covariance matrix and within-class covariance matrix, respectively. The projected data can then be compared with a threshold for assigning the corresponding predicted class, that is, predict x_* as positive class if $\boldsymbol{w}^T \boldsymbol{x}_*$ is equal or greater than the threshold; otherwise negative class.

Duda and Hart (1973) and Bishop (2006) show that the least squares solution is related to the Fisher's solution. By changing positive class labels to N/N_1 and negative class labels to $-N/N_2$, where N_1 and N_2 are the number of positive class labels and the number of negative class labels, respectively, and $N = N_1 + N_2$, the sum-of-squares error function (E) and the derivatives of E are written as:

$$
\begin{aligned}
E &= \frac{1}{2} \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}_i + w_0 - y)^2 && (2.69)\\
\frac{\partial E}{\partial w_0} &= \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}_i + w_0 - y) = 0 && (2.70)\\
\frac{\partial E}{\partial \boldsymbol{w}} &= \sum_{i=1}^{N} (\boldsymbol{w}^T \boldsymbol{x}_i + w_0 - y)\boldsymbol{x}_i = 0 && (2.71)
\end{aligned}
$$

where $\boldsymbol{\mu} = \frac{1}{N}(N_1 \boldsymbol{\mu}_1 + N_2 \boldsymbol{\mu}_2)$ and equation (2.70) leads to $w_0 = -\boldsymbol{w}^T \boldsymbol{\mu}$. By substituting $-\boldsymbol{w}^T \boldsymbol{\mu}$ into equation (2.71) and using some algebra, the following equation is obtained:

$$
\left(\boldsymbol{S}_W + \frac{N_1 N_2}{N} \boldsymbol{S}_B \right) \boldsymbol{w} = N(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2). \qquad (2.72)
$$

As $\boldsymbol{S}_B \boldsymbol{w}$ is in the direction of $(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$, the Fisher's solution becomes $\boldsymbol{w} \propto \boldsymbol{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$. We will use Fisher's linear discriminant analysis based on the least squares approach in Chapter 5 as well as SVM and GP in the experiments. The weight vector $\boldsymbol{w}$ can be

determined by:

$$\boldsymbol{y} = \begin{pmatrix} \vdots \\ N/N_1 \\ \vdots \\ \\ \vdots \\ -N/N_2 \\ \vdots \end{pmatrix} \quad , \quad \hat{\boldsymbol{X}} = \begin{pmatrix} \vdots & \vdots \\ \mathbf{1} & \boldsymbol{x}_{pos}^T \\ \vdots & \vdots \\ & \\ \vdots & \vdots \\ \mathbf{1} & \boldsymbol{x}_{neg}^T \\ \vdots & \vdots \end{pmatrix} \tag{2.73}$$

$$\boldsymbol{y} = \hat{\boldsymbol{X}}\boldsymbol{w} \tag{2.74}$$

$$\boldsymbol{w} = \hat{\boldsymbol{X}}^{-1}\boldsymbol{y} \tag{2.75}$$

$$y_{thres} = 0.5\left(\frac{N}{N_1} - \frac{N}{N_2}\right) \tag{2.76}$$

where $\boldsymbol{y}$ is the column vector of class labels in which the upper N_1 rows are N/N_1 and the lower rows are $-N/N_2$. The first column of $\hat{\boldsymbol{X}}$ is a column vector in which all elements are 1 denoted by $\mathbf{1}$ and $\boldsymbol{x}_{pos}^T$ is the transpose of a submatrix consisting of all positive class data while $\boldsymbol{x}_{neg}^T$ is another transpose of a submatrix consisting of all negative class data. The test data $\boldsymbol{x}_*$ is classified as positive class when $\langle(\mathbf{1}\ \boldsymbol{x}_*^T), \boldsymbol{w}\rangle \geqslant y_{thres}$; otherwise it is classified as negative class.

2.5 Performance Measures

To quantify the performance of a classifier, there are many available performance metrics. The fundamental results from the classifier can be summarised by a confusion matrix. Based on the confusion matrix, other evaluation metrics such as ROC, precision-recall curve and F-measure can be developed for measuring the classifier performance. Some of these metrics will be used in the experiments, while the unused metrics are described here for the sake of completeness. The evaluation metrics are described as follows.

2.5.1 Confusion matrix

A confusion matrix or contingency table is a table in which each row represents the number of instances associated with the actual class and each column shows the number of instances associated with the predicted class or vice versa. An example of a confusion matrix is shown in Table 2.2. The diagonal entries show the number of correctly classified test data while the off-diagonal entries present the number of misclassified test data by the classifier.

The following terms are information obtained by the above confusion matrix.

TABLE 2.2: An example of a confusion matrix

		Predicted Class	
		Positive	Negative
Actual Class	Positive	a	b
	Negative	c	d

- a is the number of correct predictions that a test instance is positive, which is also known as true positive (TP).
- b is the number of incorrect predictions that a test instance is negative, which is also known as false negative (FN) or Type II error.
- c is the number of incorrect predictions that a test instance is positive, which is also known as false positive (FP), false alarm, or Type I error.
- d is the number of correct predictions that a test instance is negative, which is also known as true negative (TN).
- Accuracy rate (Acc) is the proportion of the total number of correct predictions:

$$Acc = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+FN+FP+TN}. \tag{2.77}$$

- Recall or true positive rate (TPR) is the proportion of positive cases that are correctly identified:

$$TPR = \frac{a}{a+b} = \frac{TP}{TP+FN}. \tag{2.78}$$

- False positive rate (FPR) is the proportion of negative cases that are incorrectly classified as positive:

$$FPR = \frac{c}{c+d} = \frac{FP}{FP+TN}. \tag{2.79}$$

- False negative rate (FNR) is the proportion of positive cases that are incorrectly classified as negative:

$$FNR = \frac{b}{b+a} = \frac{FN}{FN+TP}. \tag{2.80}$$

- True negative rate (TNR) or specificity is defined as the proportion of negative cases that are classified correctly:

$$TNR = \frac{d}{d+c} = \frac{TN}{TN+FP} = 1 - FPR. \tag{2.81}$$

- Precision is the proportion of positive predicted cases that are true positive:

$$Precision = \frac{a}{a+c} = \frac{TP}{TP+FP}. \tag{2.82}$$

The accuracy rate calculated from the confusion matrix is a commonly used performance measure for classifiers, and we used the accuracy rate in percentage as the performance measure of classifiers in the experiments outlined in Chapter 5 and 6. The error rate is equal to (1− accuracy rate), and in Chapter 4 we reported our results of the experiments in error rate compared with other results of a published work in which the classifiers were measured by the error rate. The other aforementioned terms can also be used and developed to measure other aspects of classifiers. For example, TPR and FPR together form a metric known as ROC.

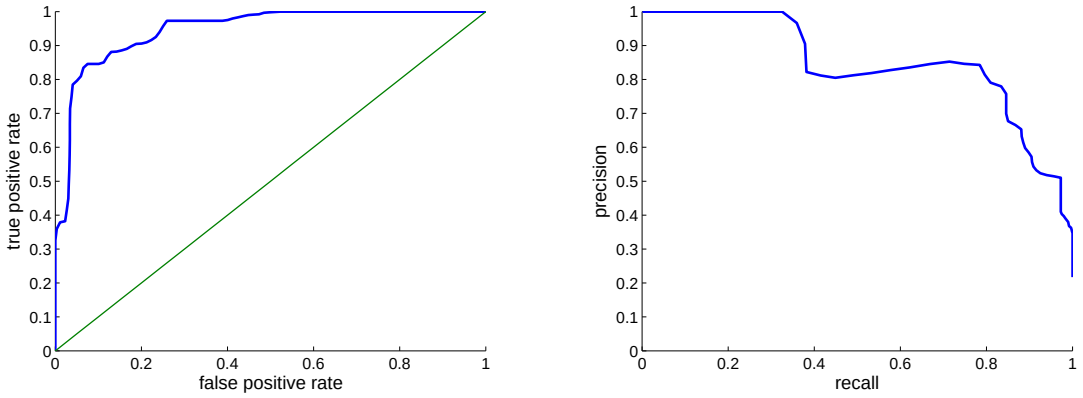
2.5.2 ROC & AUC

The receiver operating characteristic (ROC) curve (Fawcett, 2003; Provost and Fawcett, 1997) is commonly used for measuring and comparing the performance of binary classifiers. It is the graph between the true positive rate or sensitivity on the y-axis and false positive rate or (1− specificity) on the x-axis. The graph is produced by varying thresholds on the classifier outputs. The left-hand side of Figure 2.8 shows an example of the ROC curve. A classifier which has an ROC curve as the diagonal line $y = x$ behaves as a randomly guessing classifier. The closer the graph to the top left corner indicates a better classifier performance. However, it is difficult to determine which classifier performs better if there are crosses between the two ROC curves of two classifiers.

Swets (1988) proposes a convenient way of comparing the classifiers by using the area under the ROC curve (AUC). AUC represents a general behaviour of the classifier because it is independent to the threshold used for obtaining a class label. The ideal classifier has an area of 1, whereas the poor has an area of 0.5 or less. It is a widespread measure of the overall diagnostic performance and has a practical relevant interpretation as the probability of a correct discrimination in a pair of randomly selected representatives of each class (Bamber, 1975; Hanley and McNeil, 1982). Hand and Till (2001) show that the AUC is equivalent to the probability that the classifier will rank a randomly chosen member of positive class instance higher than a randomly chosen member of negative class instance. We also used AUC as the performance measure in Chapter 6.

2.5.3 Precision-recall curve

Precision and recall are widely used for defining the behaviour of an information retrieval system, for example, in search engines to measure how well a search performs. Precision is defined as the proportion of relevant documents in the set of all documents returned by the search, whereas recall is defined as the ratio of the number of relevant documents retrieved to the number of all relevant documents. Figure 2.9 illustrates the concept of precision and recall in the retrieval system. Region A (cyan-blue) and region C (purple) together represent the set of relevant documents, whereas the region B (green) together



(a) ROC curve (b) precision-recall curve

FIGURE 2.8: Examples of ROC and PR curves

with region C is the set of retrieved documents. Hence, the precision and recall can be computed as follows:

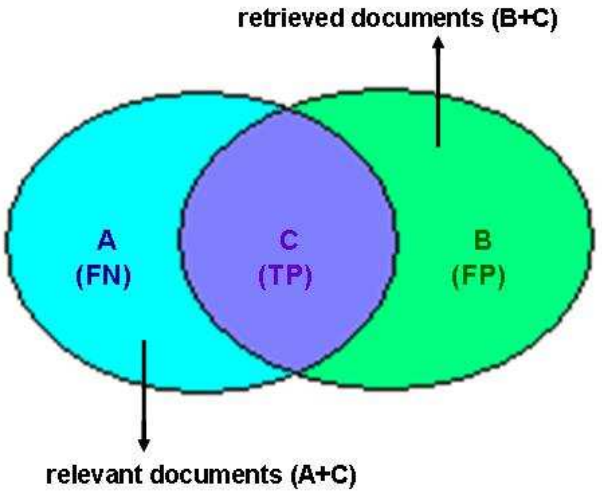


FIGURE 2.9: Diagram of the precision-recall concept in information retrieval

$$\begin{aligned} \text{precision} &= \frac{\text{number of relevant documents retrieved}}{\text{number of retrieved documents}} \\ &= \frac{C}{B+C} = \frac{TP}{FP+TP} \end{aligned} \tag{2.83}$$

$$\begin{aligned} \text{recall} &= \frac{\text{number of relevant documents retrieved}}{\text{number of relevant documents}} \\ &= \frac{C}{A+C} = \frac{TP}{FN+TP}. \end{aligned} \tag{2.84}$$

Precision-recall curve (PR curve) is the graph between precision on the y-axis and recall on the x-axis. The right pane of Figure 2.8 shows the PR curve corresponding to the ROC

curve on the left-hand side. The PR curve closest to the upper right corner indicates the best performance. Similar to AUC, the area under the PR curve can be calculated for summarising the performance by a single number, and it is related to another term known as "average precision", which is the average of the precision at each relevant document retrieved, normally cut off at the recall from 0 to 1, increasing by 0.1; hence, the precisions from these 11 points are averaged.

2.5.4 F-measure

F-measure is another evaluation in information retrieval and can be used to measure the performance of classifiers. It is sometimes referred to as F-score in statistics and can be visualised as the normalisation of region C in Figure 2.9. In other words, it measures the degree of intersection between the relevant and retrieved documents. The score is one when the relevant set is identical to the retrieval set and zero if the two sets do not overlap. It is commonly computed by:

$$\begin{aligned} F &= \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \\ &= \frac{2 \cdot (\text{precision} \cdot \text{recall})}{\text{precision} + \text{recall}} \\ &= \frac{2 \cdot TP}{FN + FP + 2 \cdot TP}. \end{aligned} \tag{2.85}$$

Van Rijsbergen (1979) introduces Rijsbergen's effectiveness measure (E_β):

$$E_\beta = 1 - \frac{1 + \beta^2}{\frac{\beta^2}{recall} + \frac{1}{precision}}. \tag{2.86}$$

Hence, the F-measure from equation (2.85) can be calculated by $F_\beta = 1 - E_\beta$ where $\beta = 1$. This is known as F_1-measure, which assigns the same weight to recall and precision. If a user gives the weight on recall two times more than on precision, that is, $\beta = 2$, the measure is called F_2-measure.

Although there are many metrics to measure classifier performance, all of them have their own weaknesses and are mostly applied to two-class problems. Only a few of them can be used directly to multi-class problems, as they may be misleading due to a skewed class distribution and unequal classification error costs. Many efforts extending ROC and AUC to multi-class problems have been developed such as Fawcett (2006) and Hand and Till (2001). However, it requires more complexity when the number of classes is very high. Amongst the aforementioned metrics, the accuracy rate is one of the metrics that many researchers generally use as an indicator of performance on multi-class problems.

2.6 Summary

This chapter reviewed some of the basic pattern classification architectures and performance measures required for the remainder of the dissertation. We restricted our focus to GPs and SVMs, which are the high performance classifications methods of choice in modern machine learning literature. We have also included FLDA as the baseline classifier in subsequent comparisons. We have further surveyed multi-class settings: OVA, OVO, DAG, UDT and ECOC. Of these, UDT is a novel architecture co-developed and introduced by this author during the early stages of his PhD. A further enhancement to this particular architecture is considered later in Chapter 6 in Section 6.4. The performance measures used to evaluate the classifiers, ROC, AUC and PR curve, are also reviewed in this chapter. Knowledge that is not directly related to classification, but applied in this thesis, for example, Markov chain Monte Carlo, kernel density estimation and bootstrap, will be explained in the corresponding chapter.

Chapter 3

Hypothesis Testing

This chapter presents and recalls certain parametric and non-parametric statistical tests, as they are widely used, and some of them were applied to evaluate the performance comparisons in this thesis. Later, we point out what is missing when the statistical tests are selected to evaluate the results of the performance comparisons in a classification context. A statistical test is a common step in data analysis and is generally known as a statistical hypothesis test or hypothesis testing. In a classification context, hypothesis testing is normally used when the performances of classifiers are compared to two or more classifiers on a number of datasets in order to determine whether a classifier statistically outperforms another one. To make the decision, two hypotheses called a null hypothesis and alternative hypothesis are set, and the hypothesis testing evaluates whether the null hypothesis should be supported or rejected. The null hypothesis, symbolized as H_0, is formally expressed in terms of "there being no difference". For example, there is no difference in performance between classifier A and classifier B denoted by $H_0 : A = B$. On the contrary, the alternative hypothesis denoted by H_1 is expressed as "there being some difference between them", which would be one of the forms $H_1 : A \neq B$, $H_1 : A > B$ or $H_1 : A < B$. They indicate whether the performance of classifier A is different, better or worse than the performance of classifier B, respectively. The first form is the non-directional alternative hypothesis known as the two-tailed or two-sided test, while the remaining form is the directional hypothesis called the one-tailed or one-sided test. There is a possibility that classifier A will perform equivalently to classifier B by chance. In order to have confidence, to some degree, that the performance of classifier A is really better or worse than that of classifier B, the null hypothesis must be shown as likely to be wrong and then nullify the null hypothesis and accept the alternative hypothesis instead. The alternative hypothesis is, however, assumed to be wrong as long as much evidence exists to the contrary. This depiction is similar to the justice system in which a suspicious person is presumed innocent rather than guilty without strong evidence. Precisely, the hypothesis testing evaluates how likely the difference would occur by chance alone if the null hypothesis were true. The previous statement is

quantified by a term called "p value", which has a value between zero and one. Hence, the p value measures how likely the observed difference is due to chance. A p value close to zero indicates that the observed difference most likely did not occur by chance, whereas a p value close to one implies that the observed difference happened due to chance. Two kinds of mistakes could happen when the hypothesis testing is carried out, referred to as a type I error (α) and type II error (β). The first kind rejects the null hypothesis when it is true, whereas the second kind does not reject the null hypothesis when it is false. Statisticians normally consider the type I error seriously by setting the p value level at 0.05. This level is called a 5 percent significance level; in other words, the observed difference due to chance can not exceed $\alpha = 5$ percent. This is the trade-off between α and β, since as α increases, β decreases and vice versa.

Although there are many tests to evaluate the null hypothesis in statistics, this chapter will only explain some common tests, starting with the parametric tests and followed by the non-parametric tests in subsequent sections. Examples in the following tests may not directly relate to the classification problem. However, in some tests, classification problem example will be used where appropriate.

3.1 Parametric Tests

Parametric in the sense of "parametric test" means that the tests require some parameters from the population distribution such as mean and variance of the distribution. Therefore, the parametric tests implicitly require some properties of the distribution. For instance, data are required to be normally distributed. One of the most widely used parametric tests between two groups is the t-test (Student, 1908). Given that there are many types of t-tests available, the choice of usage depends on the condition of each test. For more than two groups, the generalized t-test is known as ANOVA and was introduced by Fisher (1928). We describe the two-sample t-test, paired t-test and repeated-measured ANOVA in the following subsections.

3.1.1 Two-independent sample t-test

The two-independent sample t-test, sometimes called an unpaired t-test, assumes that the data are sampled from normally distributed populations in which variances are equal. The data in the two samples must be independent; in other words, these two samples are not related to each other. This is a reason why one should not use the two-independent sample t-test in classification problems given that the comparison between two classifiers usually carries on the same datasets, that is, the two samples are not independent. The more suitable t-test is the paired t-test which is described subsequently. The two-independent sample t-test determines whether the means of two independent

populations are statistically different from each other. The null hypothesis is that there is no difference between the means, whereas the alternative hypothesis is that there is a difference between the means for the two-tailed test or the mean of a population is larger than another one for the one-sided test. By convention, the 5 percent significance level is used, and if the p value is less than 0.05, the null hypothesis is rejected.

The two-independent sample t-test calculates the test statistic called the t value and the degrees of freedom (df) using the following:

$$t = \frac{\bar{X}_A - \bar{X}_B}{S_p \cdot \sqrt{\frac{1}{N_A} + \frac{1}{N_B}}} \tag{3.1}$$

$$S_p = \sqrt{\frac{(N_A - 1)S_A^2 + (N_B - 1)S_B^2}{(N_A - 1) + (N_B - 1)}} \tag{3.2}$$

$$df = (N_A - 1) + (N_B - 1) \tag{3.3}$$

where $\bar{X}$, S and N are the mean, standard deviation and the number of members in the sample, respectively. The subscripts A and B denote the associated sample. The calculated t value is compared to the critical t value associated with the degrees of freedom. The null hypothesis is rejected if the calculated t value is greater than the corresponding critical t value, or equivalently, the p value of the calculated t value is less than 0.05. A simple example is used to demonstrate the two-independent sample t-test and evaluate whether a new vitamin A has more effect on the weight/height ratio of children at the same age than vitamin B. Vitamin A was given to a group of 12 children referred to as group A, whereas another group of 12 children called group B took vitamin B. Table 3.1 shows the ratio of weight to height of both groups, including the corresponding mean and standard deviation at the bottom rows. In this example, the null hypothesis and alternative hypothesis are $H_0 : A = B$ and $H_1 : A > B$, respectively. The calculated t value is:

$$\begin{aligned} S_p &= \sqrt{\frac{(12-1)\cdot(2.26)^2 + (12-1)\cdot(3.49)^2}{(12-1)+(12-1)}} \\ &= 2.94 \\ t &= \frac{23.77 - 22.08}{2.94 \cdot \sqrt{\frac{1}{12} + \frac{1}{12}}} \\ &= 1.41 \end{aligned}$$

and degrees of freedom equal 22. The critical t value at $\alpha = 0.05$ is 1.72, which is greater than the calculated t value; hence, the null hypothesis cannot be rejected, that is, there

TABLE 3.1: Summary of data used as an example for the two-independent sample t-test. 12 children are given vitamin A and another 12 children are given vitamin B. The numbers represent the weight/height ratio in the unit of pound/feet. The last two rows show the corresponding mean and standard deviation of each group.

A			B		
26.48	21.89	22.76	18.45	23.93	21.91
26.45	19.18	22.82	20.41	20.89	29.05
26.44	25.61	24.79	20.35	16.81	20.13
21.92	23.73	23.17	26.20	21.19	25.60
mean: 23.77			mean: 22.08		
S: 2.26			S: 3.49		

is no significant difference between vitamin A and vitamin B at the 0.05 significance level.

3.1.2 Paired t-test

The paired t-test is another type of t-test that is suitable for cases when the same subject is used twice, sometimes referred to as repeated measures. For instance, the same student has been given two test sets known as a pre-test and post-test at different times to determine whether there is a significant difference between the scores from these test sets. It is formally used to compare the mean difference between the two populations when there is the dependence between samples. It tests whether or not the mean difference is significantly different from zero. The assumption of the paired t-test is that the distribution of the difference between two populations is Gaussian.

For example, if there are ten datasets classified by two classifiers, A and B, the results are shown in Table 3.2. If classifier A is expected to perform better than classifier B, then the hypotheses are set as $H_0 : A - B = 0;\ H_1 : A > B$, where $A - B = 0$ implies that there is no significant difference between the two classifiers. In a paired t-test, the calculated t value is the mean of the difference divided by the standard error of the differrence, and the degrees of freedom are equal to the number of subject pairs minus one as follows:

$$\begin{aligned} t &= \frac{\bar{X}_D}{\sqrt{\frac{S_D^2}{N}}} \\ &= \frac{0.70}{\sqrt{1.17^2/10}} = 1.89 \end{aligned} \tag{3.4}$$

$$\begin{aligned} df &= N - 1 \\ &= 10 - 1 = 9 \end{aligned} \tag{3.5}$$

TABLE 3.2: Summary of data used as an example for the paired t-test. The numbers shown in the second and third column are the average accuracy rate measured by classifier A and classifier B on ten datasets. The last column is the difference between the accuracy rates. The last two rows show the corresponding mean and standard deviation.

subject	A	B	D = A-B
1	90.70	89.32	1.38
2	75.14	76.21	-1.07
3	96.17	96.17	0.00
4	78.26	77.48	0.78
5	67.54	65.66	1.88
6	77.10	76.30	0.80
7	73.70	71.16	2.54
8	84.82	85.93	-1.11
9	97.92	97.23	0.69
10	95.14	94.02	1.12
mean	83.65	82.95	0.70
S	10.78	11.10	1.17

where $\bar{X}_D$ is the mean of the difference between the matched pair of sample A and sample B. The difference between A and B on each subject and $\bar{X}_D$ are shown in the fourth column of Table 3.2. The critical t value equals 1.83, whereas the calculated t value is 1.89, which is greater than the critical value. Therefore, the null hypothesis is rejected and the alternative hypothesis is accepted at the 0.05 significance level. In other words, classifier A performs better than classifier B at the 0.05 significance level. The results shown in Table 3.2 seem to agree with the conclusion. On the contrary, the conclusion would have conflicted with the results if the alternative hypothesis was in the opposite direction.

The two-independent sample t-test and paired t-test are different from each other due to the independence or dependence of the data and the conditions underlying the tests. It is obvious that the paired t-test is more consistent with the context of comparison between the two classifiers, as the same datasets should be used for both classifiers in order to make a reasonable comparison. However, there are often more than two classifiers or two samples when making the comparison. Therefore, the paired t-test should not be used directly when there are more than two samples. In next section, we describe how multiple samples can be statistically evaluated.

3.1.3 ANOVA

Analysis of variance, or ANOVA, is used to test whether there is any significant difference between the means of three or more groups. Instead of directly comparing the means, ANOVA analyses and compares the variances, as the name implies. The equivalent ANOVA to the independent t-test is between-subjects ANOVA. For the dependent samples there is a test called repeated/related/correlated-measures ANOVA or within-subjects ANOVA. ANOVA requires certain properties of the populations in which the data are sampled from Gaussian with the same variances. The procedure of ANOVA can be split into a pre-test and post-test. The purpose of the pre-test is to determine whether there is any difference among those groups. If there is a significant difference on the pre-test, the post-test, also called the post-hoc test, is run to determine which pairs of group means cause the difference. The choice of the post-test depends on the need to make all pairwise comparisons or to compare some groups with a selected group known as the control group. To demonstrate the between-subjects ANOVA, we use the score data shown in Table 3.3. Suppose 15 students are split into three groups and each group is taught by a different teacher. The scores are measured on an examination. The question is whether there is any significant difference in teaching by those teachers. To answer this question, one can carry out the between-subjects ANOVA because of the independent samples. The pre-test calculates the F-test statistic, which is the ratio of two numbers, as shown in equation (3.6). The numerator of the ratio is known as the mean square between groups ($MS_{between}$) and the denominator is the mean square within groups (MS_{within}). Both mean squares are computed by the corresponding sum of squares (SS) divided by the corresponding degrees of freedom. The following equations are used to compute the F-test statistic:

$$F = \frac{MS_{between}}{MS_{within}} = \frac{SS_{between}/df_{between}}{SS_{within}/df_{within}} \tag{3.6}$$

$$SS_{total} = \sum_{g=1}^{G}\sum_{i=1}^{N_g}(X_{ig})^2 - \frac{\left(\sum_{g=1}^{G}\sum_{i=1}^{N_g}X_{ig}\right)^2}{\sum_{g=1}^{G}N_g} \tag{3.7}$$

$$\begin{aligned} SS_{between} &= \sum_{g=1}^{G}\frac{\left(\sum_{i=1}^{N_g}X_{ig}\right)^2}{N_g} - \frac{\left(\sum_{g=1}^{G}\sum_{i=1}^{N_g}X_{ig}\right)^2}{\sum_{g=1}^{G}N_g} \\ &= \sum_{g=1}^{G}\frac{(C_g)^2}{N_g} - \frac{(GT)^2}{\sum_{g=1}^{G}N_g} \end{aligned} \tag{3.8}$$

$$SS_{within} = SS_{total} - SS_{between} \tag{3.9}$$

$$df_{between} = G - 1 \tag{3.10}$$

$$df_{within} = \sum_{g=1}^{G}N_g - G = N - G \tag{3.11}$$

TABLE 3.3: Summary of data used as an example for between-subjects ANOVA. Fifteen students are separated into three groups with different teachers. The numbers represent the exam marks after completing a course taught by the teachers.

	T_1	T_2	T_3	Row total
	64	42	57	163
	62	45	63	170
	51	52	52	155
	57	44	61	162
	65	42	66	173
column total	$C_1 = 299$	$C_2 = 225$	$C_3 = 299$	Grand Total GT = 823
$n = \#$data	5	5	5	total $N = 15$
mean	$\bar{T}_1 = 59.80$	$\bar{T}_2 = 45.00$	$\bar{T}_3 = 59.80$	

, where G is the number of groups and N_g is the number of data in group g. X_{ig} is the ith data of group g, in which C_g is the sum of all data, whereas GT is the grand total of all Cg, and N is the total number of data. If the calculated F-value is greater than the tabulated critical F-value with degrees of freedom $(G-1, N-G)$, the null hypothesis $H_0 : \mu_{T_1} = \mu_{T_2} = \mu_{T_3}$ is rejected. The following steps are calculated for the between-subjects ANOVA:

1. Calculate the first term of the right hand side of equation (3.7):
   $\sum X^2_{total} = 64^2 + 62^2 + 51^2 + 57^2 + 65^2 + \ldots + 57^2 + 63^2 + 52^2 + 61^2 + 66^2 = 46207$
2. Calculate the second term of the right hand side of equation (3.7):
   $\frac{GT^2}{N} = \frac{(299 + 225 + 299)^2}{5 + 5 + 5} = \frac{823^2}{15} = 45155.27$
3. $SS_{total} = 46207 - 45155.27 = 1051.73$
4. Calculate $SS_{between}$ and its degrees of freedom:

$$\begin{aligned} SS_{between} &= \frac{C_1^2 + C_2^2 + C_3^2}{n} - \frac{GT^2}{N} \\ &= \frac{299^2 + 225^2 + 299^2}{5} - 45155.27 \\ &= 45885.40 - 45155.27 \quad = \quad 730.13 \\ df_{between} &= G - 1 = 3 - 1 = 2 \end{aligned}$$

5. Calculate SS_{within} and its degrees of freedom:

$$\begin{aligned} SS_{within} &= SS_{total} - SS_{between} \\ &= 1051.73 - 730.13 = 321.60 \\ df_{within} &= N - G = 15 - 3 = 12 \end{aligned}$$

6. Calculate F-value and its degrees of freedom:

$$\begin{aligned} F &= \frac{SS_{between}/df_{between}}{SS_{within}/df_{within}} \\ &= \frac{730.13/2}{321.60/12} = 13.62 \\ df &= (df_{between}, df_{within}) = (2, 12). \end{aligned}$$

The calculated F-value is compared with the critical F-value specified by the degrees of freedom $(2, 12)$. The critical F-value is 3.89, while the calculated F-value is 13.62. The null hypothesis is then rejected at the 0.05 significance level because the calculated F-value is larger than the critical F-value. Next, any post-hoc test described below can be used to determine which pairs of means are significantly different from each other when all pairwise comparisons are of interest.

The first post-hoc test is known as Fisher's least significant difference (LSD) by Fisher (1935). It makes all pairwise comparisons of the mean differences and compares them to the LSD value, which is computed by the following:

$$\begin{aligned} LSD &= t\sqrt{\frac{2MS_{within}}{n}} \\ &= 2.18\sqrt{\frac{2(321.60/12)}{5}} \\ &= 7.14 \end{aligned} \tag{3.12}$$

, where t is the critical t value of the t-distribution with df_{within}. In this example, the critical t value at $\alpha = 0.05$ and $df_{within} = 12$ is 2.18; hence, the LSD equals 7.14. Subsequently, the absolute value of each pairwise difference in the means is compared to the LSD value. If the absolute value of any pairs is greater than the LSD value, those pairs are significantly different at the α level. The results of the Fisher's LSD show that T_1 and T_3 differ from T_2 as follows:

$$\begin{aligned} |\bar{T}_1 - \bar{T}_2| = 14.80 &\geqslant 7.14 \quad \text{reject} \quad H_0 : \mu_{T_1} = \mu_{T_2} \\ |\bar{T}_1 - \bar{T}_3| = \ 0.00 &< 7.14 \quad \text{do not reject} \quad H_0 : \mu_{T_1} = \mu_{T_3} \\ |\bar{T}_2 - \bar{T}_3| = 14.80 &\geqslant 7.14 \quad \text{reject} \quad H_0 : \mu_{T_2} = \mu_{T_3}. \end{aligned}$$

Tukey's HSD test (Tukey, 1949) is another post-hoc test and has a similar equation to Fisher's LSD. However, it is based on the Studentized range distribution rather than

the t-distribution. It calculates the following:

$$\begin{aligned} HSD &= q_{(\alpha,k)}\sqrt{\frac{MS_{within}}{n}} \\ &= 3.77\sqrt{\frac{(321.60/12)}{5}} \\ &= 8.73 \end{aligned} \tag{3.13}$$

, where k is the number of groups and $q_{(\alpha,k)}$ refers to the Studentized range statistic table with df_{within} at the corresponding α level. For the present example $q_{(0.05,3)}$ is 3.77, so the HSD equals 8.73. Similar to Fisher's LSD, any pair with a mean difference greater than the HSD value will nullify the null hypothesis. The Tukey's HSD test shows the same conclusion as Fisher's LSD test as follows:

$$\begin{aligned} |\bar{T}_1 - \bar{T}_2| = 14.80 &\geqslant 8.73 \quad \text{reject} \quad H_0 : \mu_{T_1} = \mu_{T_2} \\ |\bar{T}_1 - \bar{T}_3| = \ 0.00 &< 8.73 \quad \text{do not reject} \quad H_0 : \mu_{T_1} = \mu_{T_3} \\ |\bar{T}_2 - \bar{T}_3| = 14.80 &\geqslant 8.73 \quad \text{reject} \quad H_0 : \mu_{T_2} = \mu_{T_3}. \end{aligned}$$

The next pairwise post-hoc test illustrated here is Scheffé's test (Scheffé, 1953). Its procedure is similar to the previous two post-hoc tests, but the formula is:

$$\begin{aligned} CD &= \sqrt{(k-1)F_\alpha\left(\frac{2MS_{within}}{n}\right)} \\ &= \sqrt{(3-1)(3.89)\left(\frac{2(321.60/12)}{5}\right)} \\ &= 9.13 \end{aligned} \tag{3.14}$$

, where k is the number of groups and F_α is the critical F-value with the degrees of freedom $(df_{between}, df_{within})$. Any pair with a mean difference greater than the critical difference (CD) will nullify the null hypothesis. The Scheffé's test shows that:

$$\begin{aligned} |\bar{T}_1 - \bar{T}_2| = 14.80 &\geqslant 9.13 \quad \text{reject} \quad H_0 : \mu_{T_1} = \mu_{T_2} \\ |\bar{T}_1 - \bar{T}_3| = \ 0.00 &< 9.13 \quad \text{do not reject} \quad H_0 : \mu_{T_1} = \mu_{T_3} \\ |\bar{T}_2 - \bar{T}_3| = 14.80 &\geqslant 9.13 \quad \text{reject} \quad H_0 : \mu_{T_2} = \mu_{T_3} \end{aligned}$$

which agrees with the conclusion of the previous post-hoc test. Sometimes all pairwise comparisons are unnecessary if an experimenter only needs to compare all groups to a reference or control group. Dunnett's test (Dunnett, 1980) is designed for this purpose.

TABLE 3.4: Summary of data used as an example for repeated-measures ANOVA. Five workers operate three brands of machines producing the same products. The numbers in columns 2, 3 and 4 are the operating time (minutes) of the corresponding machines.

subject	M_1	M_2	M_3	Row total
1	5	8	7	$P_1 = 20$
2	6	10	8	$P_2 = 24$
3	4	6	4	$P_3 = 14$
4	7	9	10	$P_4 = 26$
5	3	8	5	$P_5 = 16$
column total	$C_1 = 25$	$C_2 = 41$	$C_3 = 34$	Grand Total GT = 100
n = #data	5	5	5	total $N = 15$
mean	$\bar{M}_1 = 5$	$\bar{M}_2 = 8.2$	$\bar{M}_3 = 6.8$	

It calculates:

$$t_{d(i,c)} = \frac{M_i - M_c}{\sqrt{\frac{2MS_{within}}{n}}} \tag{3.15}$$

, where M_c is the mean of the control group and M_i is the mean of the ith group. The calculated t_d value is compared with the critical value of Dunnett's table with degrees of freedom $(N - k)$ where N is the total number of data in all groups and k is the number of data groups. If T_1 is the control group, the Dunnett's test shows that:

$$t_{d(T_2,T_1)} = \frac{59.80 - 45.00}{\sqrt{\frac{2(321.60/12)}{5}}} = 4.52$$

$$t_{d(T_3,T_1)} = \frac{59.80 - 59.80}{\sqrt{\frac{2(321.60/12)}{5}}} = 0$$

where the critical Dunnett value at the 0.05 significance level in this example is 2.5. Therefore, T_2 significantly differs from T_1 at the 0.05 level, as $t_{d(T_2,T_1)} = 4.52 > 2.5$.

So far, we have described between-subjects ANOVA, which assumes data independence. In the case of dependent groups, e.g., same subject under different conditions, repeated-measures ANOVA is a suitable hypothesis testing. The data in Table 3.4 are used to demonstrate the repeated-measures ANOVA. In this example, there are three brands of machines producing the same products and five workers are selected to test the machines. The company wants to know if there is any difference in operating time of these machines, so the repeated-measures ANOVA is performed. The operating time is measured and shown in the corresponding columns. The procedure of repeated-measures ANOVA is

similar to between-subjects ANOVA, but the F-value is calculated as follows:

$$F = \frac{MS_{between}}{MS_{error}} = \frac{SS_{between}/df_{between}}{SS_{error}/df_{error}} \tag{3.16}$$

$$SS_{total} = \sum_{g=1}^{G}\sum_{s=1}^{S}(X_{sg})^2 - \frac{GT^2}{N} \tag{3.17}$$

$$SS_{between} = \sum_{g=1}^{G}\frac{C_g^2}{S} - \frac{GT^2}{N} \tag{3.18}$$

$$SS_{subject} = \sum_{s=1}^{S}\frac{P_s^2}{G} - \frac{GT^2}{N} \tag{3.19}$$

$$SS_{error} = SS_{total} - SS_{between} - SS_{subject} \tag{3.20}$$

$$df_{between} = G - 1 \tag{3.21}$$

$$df_{error} = (G-1)(S-1) \tag{3.22}$$

, where G is the number of groups and S is the number of subjects. X_{sg} is the data of subject s in group g, in which C_g is the sum of all data. P_s is the sum of all data in the same subject s, while GT is the grand total of all Cg, and N is the total number of data. If the calculated F-value is greater than the tabulated critical F-value with degrees of freedom $(G-1, (G-1)(S-1))$, the null hypothesis is rejected. The following steps are calculated for repeated-measures ANOVA:

1. Calculate $\sum X_{total}^2 = 5^2 + 6^2 + 4^2 + 7^2 + 3^2 + \ldots + 7^2 + 8^2 + 4^2 + 10^2 + 5^2 = 734$
2. Calculate $\frac{GT^2}{N} = \frac{100^2}{15} = 666.67$
3. $SS_{total} = 734 - 666.67 = 67.33$
4. Calculate $SS_{between}$ and its degrees of freedom:

$$\begin{aligned} SS_{between} &= \frac{C_1^2 + C_2^2 + C_3^2}{S} - \frac{GT^2}{N} \\ &= \frac{25^2 + 41^2 + 34^2}{5} - 666.67 \\ &= 692.40 - 666.67 \quad = \quad 25.73 \\ df_{between} &= G - 1 = 3 - 1 = 2 \end{aligned}$$

5. Calculate $SS_{subject}$:

$$\begin{aligned} SS_{subject} &= \frac{P_1^2 + P_2^2 + P_3^2 + P_4^2 + P_5^2}{G} - \frac{GT^2}{N} \\ &= \frac{20^2 + 24^2 + 14^2 + 26^2 + 16^2}{3} - 666.67 \\ &= 701.33 - 666.67 \quad = \quad 34.66 \end{aligned}$$

6. Calculate SS_{error} and its degrees of freedom:

$$\begin{aligned} SS_{error} &= SS_{total} - SS_{between} - SS_{subject} \\ &= 67.33 - 25.73 - 34.66 = 6.94 \\ df_{error} &= (G-1)(S-1) = (2)(4) = 8 \end{aligned}$$

7. Calculate F-value and its degrees of freedom:

$$\begin{aligned} F &= \frac{SS_{between}/df_{between}}{SS_{error}/df_{error}} \\ &= \frac{25.73/2}{6.94/8} = 14.83 \\ df &= (df_{between}, df_{error}) = (2, 8). \end{aligned}$$

The critical F-value with degrees of freedom $(2, 8)$ is 4.46. The null hypothesis is rejected at the 0.05 significance level because the calculated $F = 14.83$ is larger than the critical F-value. To determine which pairs cause the difference, any aforementioned post-hoc test can be used, but replacing MS_{within} with MS_{error} in the corresponding post-hoc test. For example, the Scheffé's test becomes:

$$\begin{aligned} CD &= \sqrt{(k-1)F_\alpha \left(\frac{2MS_{error}}{n}\right)} \\ &= \sqrt{(3-1)(4.46)\left(\frac{2(6.94/8)}{5}\right)} \\ &= 1.76 \end{aligned} \tag{3.23}$$

and it shows that M_1 differs from M_2 and M_3 as follows:

$$\begin{aligned} |\bar{M}_1 - \bar{M}_2| = 3.2 &\geqslant 1.76 \quad \text{reject} \quad H_0 : \mu_{M_1} = \mu_{M_2} \\ |\bar{M}_1 - \bar{M}_3| = 1.8 &\geqslant 1.76 \quad \text{reject} \quad H_0 : \mu_{M_1} = \mu_{M_3} \\ |\bar{M}_2 - \bar{M}_3| = 1.4 &< 1.76 \quad \text{do not reject} \quad H_0 : \mu_{M_2} = \mu_{M_3}. \end{aligned}$$

The above repeated-measure ANOVA is formally referred to as one-way repeated-measures ANOVA, and it is a suitable parametric test as long as the required assumptions are fulfilled. Other types of ANOVA can be found in many statistical books, but they

are not presented here, as they are not usually applied or relevant to the classification context. Nonetheless, if the assumptions of the parametric tests are invalid, then the non-parametric tests would be preferred to the parametric ones.

3.2 Non-Parametric Tests

The parametric tests assume the normality property; hence, before using them, a test of normality such as Shapiro-Wilk (Shapiro and Wilk, 1965) or Kolmogorov-Smirnov test (Massey, 1951) could be used for checking the normality. However, the normality tests usually fail for small sample sizes. In contrast to the parametric tests, the non-parametric tests do not depend on the normality of underlying populations, and they can be applied to the same data used by the parametric tests as long as the data can be ranked. Another difference is that the non-parametric tests compare medians rather than means, as in the parametric tests. The non-parametric test known as the Mann-Whitney U test, the Wilcoxon matched-pairs signed-ranks test and the Friedman test are presented, as they are the counterparts of the parametric tests in the previous section.

3.2.1 Mann-Whitney U test

The Mann-Whitney U test (Mann and Whitney, 1947) is a non-parametric test that is equivalent to the two-sample independent t-test. The test evaluates whether the two samples are sampled from two populations with the same median value. It starts with combining data from two samples and ranking them from the lowest to the highest scores, including the average rank when tie ranks occur. The data shown in Table 3.5 demonstrates the Mann-Whitney U test. There are five data for each group and the corresponding ranks, R_A and R_B, after combining the two groups, are shown in column 2 and 4, respectively. The rank sum $\sum R$ and average rank sum $\sum \bar{R}$ of each group are also shown at the bottom of Table 3.5. Then the Mann-Whiteny U test calculates the U value of each group: U_A and U_B, by:

$$\begin{aligned} U_A &= n_A n_B + \frac{n_A(n_A+1)}{2} - \sum R_A \qquad (3.24) \\ &= (5)(5) + \frac{5(5+1)}{2} - 18 \\ &= 22 \\ U_B &= n_A n_B + \frac{n_B(n_B+1)}{2} - \sum R_B \qquad (3.25) \\ &= (5)(5) + \frac{5(5+1)}{2} - 37 \\ &= 3 \end{aligned}$$

TABLE 3.5: Summary of data used as an example of Mann-Whitney U test. R_A and R_B are the rank score of corresponding data of group A and group B, respectively after combining data from both groups. $\sum R_A$ and $\sum R_B$ are the sum of the rank scores of each group. The average rank sums are shown in the last row.

Group 1		Group 2	
A	R_A	B	R_B
8.9	8	9.4	9.5
4.6	1.5	7.9	6
4.9	3	8.2	7
5.2	4	9.4	9.5
4.6	1.5	6.2	5
	$\sum R_A = 18$		$\sum R_B = 37$
$\bar{R}_A = \frac{\sum R_A}{n_A} = \frac{18}{5} = 3.6$		$\bar{R}_B = \frac{\sum R_B}{n_B} = \frac{37}{5} = 7.4$	

where n_A and n_B are the number of data in group A and B, respectively. The smaller value between U_A and U_B is chosen as the U statistic and is compared with the critical value from the table of critical values for the Mann-Whitney U statistic at the α level. If the obtained smaller U value is less than or equal to the critical value, the null hypothesis is rejected. The one-tailed critical value at $\alpha = 0.05$ is 4, which is greater than the obtained U value. Hence, the null hypothesis can be rejected and the alternative hypothesis is supported. However, there are two cases for the alternative hypothesis that is either $H_1 : A < B$ or $H_1 : A > B$. It turns out that the alternative hypothesis $H_1 : A < B$ is supported as the evidence of $\bar{R}_A < \bar{R}_B$, while the data are not consistent with another alternative hypothesis. For a large sample size, the normal approximation of the Mann-Whitney U test can be employed by calculating (Sheskin, 2004):

$$z = \frac{U - \frac{n_A n_B}{2}}{\sqrt{\frac{n_A n_B (n_A + n_B + 1)}{12}}}. \tag{3.26}$$

The null hypothesis is rejected if the absolute of the calculated z is greater than or equal to the critical z value from the table of the normal distribution.

3.2.2 Wilcoxon matched-pairs signed-ranks test

The Wilcoxon matched-pairs signed-ranks test (Wilcoxon, 1945) is the counterpart of the paired t-test, but it is not restricted to the normality and variance equality. An example of a classification problem that consists of ten datasets classified by classifier A and B is used to illustrate the test. We summarise the data for this example at the top of Table 3.6. In column 4, D is the difference between mean scores from both classifiers.

In column 5, the D scores are ranked with respect to their absolute values, while column 6 lists the signed rank of the D scores. A lower rank is assigned to the smaller absolute value, and the average rank is assigned to the case of ties. Any row with a difference score of zero is not ranked. The sum of the positive and negative ranks denoted by $\sum R_+$ and $\sum R_-$ are shown at the bottom of column 6. If the value of $\sum R_+$ is significantly greater than the value of $\sum R_-$, it indicates that classifier A is likely to perform better than classifier B and vice versa. The null hypothesis ($H_0 : A = B$) can be rejected if the smaller value $T = \min(\sum R_+, \sum R_-)$ is less than or equal to the tabled critical value at the pre-defined level of significance (T_α). By convention $\alpha = 0.05$ is used for the significance level.

TABLE 3.6: Summary data used as examples for the Wilcoxon matched-pairs signed-ranks test (top) and the Friedman test (bottom). Ten classification problems are classified by classifiers. The scores are the average accuracy rate by the corresponding classifiers. At the bottom, the numbers in parentheses are the rank scores of corresponding classifiers and the last row shows the calculated F_F value and the critical F-value.

	dataset	A	B	$D = A - B$	Rank of $\|D\|$	Signed rank of $\|D\|$
Wilcoxon	1	97.13	91.82	5.31	5	5
	2	97.66	97.60	0.06	1	1
	3	78.14	70.35	7.79	9	9
	4	82.59	76.94	5.65	6	6
	5	89.43	88.70	0.73	2	2
	6	87.99	82.03	5.96	8	8
	7	69.38	68.48	0.90	3	3
	8	98.32	92.51	5.81	7	7
	9	69.37	69.37	0.00	-	-
	10	70.00	72.78	-2.78	4	-4
						$\sum R_+ = 41$
						$\sum R_- = 4$

	dataset	A	B	C
Friedman	1	97.13(1)	91.82(2)	90.76(3)
	2	97.66(2)	97.60(3)	98.82(1)
	3	78.14(1)	70.35(3)	71.99(2)
	4	82.59(1)	76.94(3)	80.09(2)
	5	89.43(1)	88.70(2)	82.21(3)
	6	87.99(1)	82.03(2)	80.35(3)
	7	69.38(1)	68.48(2)	65.35(3)
	8	98.32(1)	92.51(2)	90.76(3)
	9	69.37(1.5)	69.37(1.5)	66.59(3)
	10	70.00(3)	72.78(1)	72.56(2)
	averge rank	$\bar{R}_A$=1.35	$\bar{R}_B$=2.15	$\bar{R}_C$=2.5
	$F_{F(2,18)}$	= 4.79,	$F_{0.05,2,18}$	= 4.41

The absolute value of the smaller value T is known as the Wilcoxon T-test statistic. Given that $\sum R_- = 4$ is less than $\sum R_+ = 41$, T equals 4. In order for the directional alternative hypothesis ($H_1 : A > B$) to be accepted, $\sum R_+$ must be greater than $\sum R_-$, and T must be less than or equal to the critical one-tailed $T_{0.05} = 8$. Since the Wilcoxon $T = 4$ is less than the critical $T_{0.05} = 8$, the alternative H_1 is supported at the 0.05 significance level. It should be noted that for a larger number of datasets (M), for example $M > 25$, the z statistic can be computed by (Sheskin, 2004):

$$z = \frac{T - \frac{1}{4}M(M+1)}{\sqrt{\frac{1}{24}M(M+1)(2M+1)}} \tag{3.27}$$

which is compared with the tabled critical one-tailed at the 0.05 significance level $z_{0.05} = 1.65$. If the absolute obtained value z is greater than 1.65, the null hypothesis H_0 is rejected and the alternative hypothesis H_1 is accepted. For a smaller number of datasets, the critical T value can be determined by the formula (McCornack, 1965):

$$Pr[T \leqslant t|M] = \frac{\sum_{T=0}^{t} F_{T,M}}{2^M} \tag{3.28}$$

$$F_{t,M} = F_{t,M-1} + F_{t-M,M-1} \tag{3.29}$$

, where $F_{t,M}$ is the number of ways that the positive ranks (negative ranks if the total sum of the negative-signed rank is less than the total sum of the positive-signed rank) sum to t when M data are ranked and $F_{t-M,M-1} = 0$ if $(t - M) < 0$. For example, for $t = 3; M = 3$, there are $2^3 = 8$ possible ways of sum, but there are only two ways whose sum equals 3, that is, 3 and $1 + 2$, hence, $F_{3,3} = 2$. In the same way $F_{0,3}, F_{1,3}, F_{2,3}, F_{4,3}, F_{5,3}$ and $F_{6,3}$ are all equal to one.

3.2.3 Friedman test

The Friedman test (Friedman, 1940) is a non-parametric test for conducting multiple comparisons on multiple groups. Hence, it is equivalent and an alternative to the repeated-measures ANOVA. As an example shown at the bottom of Table 3.6, in which three classifiers run on 10 datasets, the Friedman test ranks the scores of each classifier on the same datasets from the highest score (lowest rank) to the lowest score (highest rank). Similar to the ANOVA, there are two steps of the Friedman test: pre-test and post-hoc test. The pre-test is used to determine whether there is any statistically different performance among the multiple groups (classifiers) under H_0 : all classifiers perform equivalently. If H_0 is rejected, the post-hoc test is needed to determine which pairs of classifiers have significant differences. In this example, the numbers in column 2, 3 and 4 are the average accuracy measured by the corresponding classifier A, B and C. The numbers in parentheses indicate the rank of each classifier for the same dataset. A lower rank is assigned to the higher accuracy and the average rank is assigned in case

of a tie. For example, in the case of ties for the ninth dataset, both classifier A and B produce the same accuracy. Hence, the rank score $(1+2)/2 = 1.5$ is assigned.

The average ranks $\bar{R}_j = \frac{1}{M}\sum_{i=1}^{M} r_i^j$ are calculated, where r_i^j are the rank of the jth of k classifier on the ith of M datasets, e.g., $\bar{R}_1 = \bar{R}_A = 1.35$, $M = 10$ and $k = 3$ in this example. In the pre-test, the Friedman statistic F_F, which is distributed according to the F-distribution with $(k-1)$ and $(k-1)(M-1)$ degrees of freedom, is computed. The statistic F_F is then compared with tabled critical value F_α. If the calculated $F_F \geqslant F_\alpha$, the null hypothesis H_0 is rejected and the post-hoc test is performed. In the post-hoc step, the performances between the two classifiers are considered significantly different if the difference of the corresponding average ranks is greater than or equal to the critical difference value (CD). The following equations are used for calculating the statistic values (Demšar, 2006):

$$\begin{aligned}
\chi_F^2 &= \frac{12M}{k(k+1)}\left[\sum_j \bar{R}_j^2 - \frac{k(k+1)^2}{4}\right] \qquad (3.30)\\
&= \frac{12\times 10}{3\times 4}\left[(1.35^2 + 2.15^2 + 2.5^2) - \frac{3\times 4^2}{4}\right]\\
&= 10\times(12.695 - 12) = 6.95
\end{aligned}$$

$$\begin{aligned}
F_F &= \frac{(M-1)\chi_F^2}{M(k-1)-\chi_F^2} \qquad (3.31)\\
&= \frac{(10-1)\times 6.95}{10\times 2 - 6.95} = 4.79
\end{aligned}$$

$$\begin{aligned}
CD &= q_\alpha\sqrt{\frac{k(k+1)}{6M}} \qquad (3.32)\\
&= 2.052\times\sqrt{\frac{3(3+1)}{6(10)}} = 0.92
\end{aligned}$$

$$z_{ic} = \frac{(\bar{R}_i - \bar{R}_c)}{\sqrt{\frac{k(k+1)}{6M}}} \qquad (3.33)$$

, where q_α is based on the Studentized range statistic, with degrees of freedom $= \infty$, divided by $\sqrt{2}$, and z_{ic} is based on normal distribution with an appropriate α. The obtained value F_F is 4.79, which is greater than the tabled critical value 4.41 that is H_0 is rejected, at which point the post-hoc test is required. The equation (3.32) is used for all pairwise comparisons, so $q_{\alpha=0.05} = 2.052$ and the one-tailed critical difference value is $2.052\times\sqrt{\frac{3(3+1)}{6(10)}} = 0.92$. The differences of the average ranks between A and B, A and C, and B and C are $0.8, 1.15$, and 0.35, respectively. As the average rank of A is less than that of C, and it differs from the average rank of C by more than 0.92, classifier A

performs statistically better than classifier C at the 0.05 significance level. However, the data are not sufficient to reach any conclusion between classifier A and B, or between classifier B and C because the critical difference value is greater than the differences of these average ranks. When all classifiers are compared with a control classifier, equation (3.33) is used. If A is the control classifier, then $z_{BA} = 1.79$ and $z_{CA} = 2.57$. Using z_{ic}, one can find the corresponding p value, which is calculated by $1 - normcdf(|z_{ic}|)$, where *normcdf* is the normal cumulative distribution function. The p values of z_{BA} and z_{CA} are 0.037 and 0.0051, respectively. The appropriate α is $\alpha/(k-1)$ when comparing with the control classifier, so the α is set to $0.05/2 = 0.025$. Given that the p value of z_{BA} is greater than 0.025, the null hypothesis cannot be rejected, whereas the p value of z_{CA} is less than 0.025, the null hypothesis can be rejected. As a result, classifier A outperforms classifier C.

3.3 What's Missing

A number of issues are overlooked when the statistical tests are directly used for comparisons of classifiers. Some experimenters applied the t-test, but they ignored the validity of its underlying assumptions. Although non-parametric tests were used, the tests only consider the ranks of scores. This may cause a misleading conclusion if a classifier always has a lower rank than another classifier, while the magnitudes of score differences are small. An important issue in classification is the uncertainty in those scores when one runs a cross-validation. Normally, one runs the cross-validation or repeatedly splits a dataset into multiple training and test data, constructs the classifier on training data, and then measures the performance by averaging the scores on the test data. These average scores are then used for the statistical hypothesis test. However, the traditional hypothesis testing does not take uncertainty from cross-validation into account, and we may lose useful information from the uncertainty. For illustration purposes, Figure 3.1 shows the effect of uncertainty that could influence the decision. The A and B nodes show the average accuracy from running the cross-validation on classifier A and B, respectively, and the uncertainty from the cross-validation is represented by the horizontal line attached to the node. It is seen in the upper left of the figure that there is the difference in performance between classifier A and B, since the average accuracies are far apart and there is no overlap because the possible maximum accuracy by classifier A is less than the minimum accuracy by classifier B. In this case, the uncertainty does not have much influence due to the large gap between those average accuracies. In contrast, the remaining subfigures of Figure 3.1 show the effect of the uncertainty when the gap is smaller. When the difference between the average accuracies comes closer and the overlapping uncertainty area of the two classifiers has more intersections, the uncertainty now affects the confidence of the decision to some extent. Therefore, only

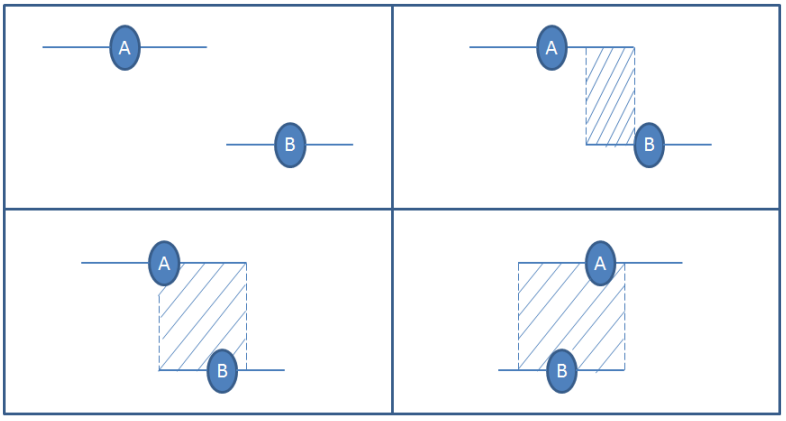


FIGURE 3.1: Illustration of the uncertainty effect obtained from the cross-validation. Nodes A and B represent the average accuracy rate from classifiers A and B, respectively. The horizontal lines attached to the nodes indicate the uncertainty of the average accuracy from the cross-validation, and the diagonal lines show the overlapping area of the uncertainty between the classifiers.

using the average accuracy in the usual way for the cross-validation could mislead the result of classifier comparison.

It will be fine if one uses traditional hypothesis testing on a large number of datasets, as the performance may be estimated towards a normal distribution by the law of large numbers and central limit theorem. In practice, however, experimenters run classifiers on a small number of datasets due to the expensive costs, such as computation and class labelling; hence, using the average scores together with uncertainties may provide benefit for hypothesis testing. To this end, we propose a number of sampling-based hypothesis testing, which considers both accuracy and uncertainty and the details are discussed in Chapter 5.

3.4 Summary

We have provided a brief review of the hypothesis testing methods used to compare the pattern classification methods. These are in the form of parametric and non-parametric tests, the former making assumptions about the functional forms of specific distributions. The t-test, most widely used in the literature, is only suitable for two-group comparisons. ANOVA is the parametric test that deals with multiple groups, and it is equivalent to the t-test when the number of groups is reduced to two. The Mann-Whitney U test, Wilcoxon matched-pairs signed-ranks test and Friedman test are non-parametric tests that are equivalent to the independent t-test, dependent t-test and repeated-measures ANOVA, respectively. Many published works omit the statistical test; hence, there is a

possibility that the satisfactory results may be caused by chance. Even though the authors have applied a statistical test, the test makes a judgment based on the average performances, e.g., average accuracies, and eliminates uncertainty from the cross-validation or multiple training-test pairs. In Chapter 5, we present new methods of sampling-based statistical hypothesis testing, which consider both accuracy and uncertainty.

Chapter 4

Meta-Analysis of GP Classifier Performance

The Gaussian process method has been used for classification for more than ten years now. While its use in regression has a long history dating back to its use in geostatistics (known as kriging) and in the statistical literature (O'Hagan, 1978), its introduction to machine learning literature following the work of Carl Rasmussen, the DELVE suite of benchmark problems and evaluations in particular, triggered a significant surge of activity. As substitutes for artificial neural networks and the radial basis function that dominated the scene during the late 1980s and early 1990s, Gaussian process methods have Bayesian probabilistic appeal, which is the ability to summarise uncertainties associated with the inference process as predictive probability densities. In short, a GP classifier is a Bayesian inference model that predicts the class label $y = \{-1, 1\}$, in two-class problems, for input $\boldsymbol{x}_*$ whose predictive distribution is obtained by the expectation of the posterior distribution over the latent function values. Recently, sparse GP approximations have been proposed, since they require fewer resources for the estimation. To avoid scaling problems, only $M \ll N$ input data points known as an active set are chosen for representing the dataset, and the running time is reduced to $O(NM^2)$ rather than $O(N^3)$ in training, and $O(M^2)$ rather than $O(N^2)$ in testing. We have provided the details of GP and sparse GP in Chapter 2 under Section 2.3 and 2.3.4, respectively.

In this chapter we provide a meta-analysis, from a practitioner's point of view, to determine whether the GP classifiers are actually competitive with other inference techniques. We were motivated by the experimental comparisons conducted on a novel sparse approximation method by Naish-Guzman and Holden (2008). While the purpose of the sparse approximation is computational saving with a minimum loss of classification accuracy, the results show that in a number of experiments, the sparse method actually outperforms the full GP formulation. Furthermore, on a number of problems considered, the sparse method achieves this with only two basis functions. This has led us

to suspect the experimental evaluations of GP classification. We begin with the results retrieved from the literature in Section 4.1 and try to draw some conclusions from the literature. In Section 4.2, we perform some experiments and compare the results with those of Naish-Guzman and Holden (2008), particularly on the two-class problems in which the sparse method used only $M = 2$.

4.1 Literature Results

We carried out an exhaustive search for Gaussian and non-Gaussian process classifications in the mainstream journals and conferences of machine learning during a ten-year period with the same reported benchmark datasets. The characteristics of these datasets are displayed in Table 4.1 where the training data size ranges from 140 to 1000, while the test data size ranges from 45 to 10000. The number of classes varies between 2 and 27 and dimensionality varies between 2 and 784. A number of datasets have a fixed training-test partition, whereas some datasets were evaluated using ten-fold cross-validation (10CV). There are certain specifics in the use of these datasets. For example, `bupa1` and `bupa2` refer to the same dataset, but they differ in the way training-test partitioning is carried out. The `wisconsin1` is the original `wisconsin` dataset. The `wisconsin2` and `wisconsin3` are diagnostic datasets that differ from the original. Some datasets such as `flare-solar1` and `flare-solar2` were evaluated with a different dimension although both are, in fact, the same. The attributes used are different, thereby making a comparison difficult.

Table 4.1: Characteristics of common benchmark datasets retrieved from the literature.

dataset	#class	#dim	train-test
banana	2	2	400-4900
breast cancer	2	9	200-77
bupa1	2	6	230-115
bupa2	2	6	345(10CV)
crabs	2	5	80-120
dna1	3	180	300-2886
dna2	3	180	2000-1186
flare-solar1	2	9	666-400
flare-solar2	2	10	1066 (10CV)
glass	6	9	214 (10CV)
German credit	2	20	700-300
heart1	2	13	170-100
heart2	2	13	270 (10CV)
iris1	3	4	150 (10CV)
iris2	3	4	105-45

Continued on next page

Table 4.1 – continued from previous page

dataset	#class	#dim	train-test
ionosphere1	2	34	351 (10CV)
ionosphere2	2	34	200-151
segmentation	7	18	1300-1010
isolet	26	617	6238-1559
MNIST1	10	169	60000-10000
MNIST2	10	784	60000-10000
pima diabetes	2	8	768(10CV)
protein fold	27	125	314-385
ringnorm	2	20	400-7000
satellite image1 (satimage1)	6	36	1000-5435
satellite image2 (satimage2)	6	36	4435-2000
sonar1	2	60	208 (10CV)
sonar2	2	60	104-104
splice	2	60	1000-2175
thyroid1	2	5	140-75
thyroid2	2	5	215 (10CV)
thyroid3	3	21	4800-2400
thyroid4	3	21	4320-2880
titanic	2	3	150-2051
twonorm	2	20	400-7000
USPS	10	256	7291-2007
waveform1	2	21	400-4600
waveform2	3	21	400-4600
waveform3	3	21	150-4850
wine	3	13	178 (10CV)
wisconsin1(original)	2	9	683 (10CV)
wisconsin2	2	30	300-269
wisconsin3	2	30	569 (10CV)

Table 4.2 shows the comparison of the classification error rate between Gaussian process and non-Gaussian process classifiers retrieved from the literature in which the full name of the abbreviation is shown separately in Table 4.4 due to the limited table width. The last column of Table 4.2 indicates the number of data items that correspond to the 1 percent error rate on the test data. In a vast majority of the cases, the results presented by the associated authors miss the crucial point about quantifying uncertainty in the results with respect to randomly partitioning the data into training and test sets. Failing to quantify this uncertainty has the danger of publishing results that are tuned to the test set. Hence, it will not be possible to determine whether the authors are reporting fortuitous "good news". When this information is absent, it is also impossible to compute a measure of statistical significance by comparing the methods.

Amongst 43 comparisons, only four problems show both corresponding means and standard deviations in both classifiers, whereas seven datasets provide only means in the GP classifier column and both means and standard deviations in the non-GP classifier column. Furthermore, only two datasets provide means in the non-GP classifier column and both means and standard deviations in the GP classifier column. Formal methods of comparing classifiers have been discussed in Demšar (2006) and Salzberg (1997). However, the methods recommended in their work cannot be applied to the results from the literature due to different settings, especially the unmatch-paired data amongst them. Furthermore, the authors do not publish accuracies of classifiers in the individual partition of the cross-validation setting. Therefore, we had to be satisfied with the use of unpaired t-tests for conducting statistical comparisons of the classifiers, even though the assumption of data independence may be invalid.

The performance of GP against non-GP classifiers on the four datasets, `dna1`, `iris1`, `ionosphere1` and `satimage1`, can be compared by the statistical unpaired t-test, since their means and standard deviations are provided. The two-tailed unpaired t-test is employed to evaluate the statistical significance of the difference between GP and non-GP classifiers at the 0.05 significance level on those four datasets. The tests show that significant differences exist between both classifiers on `dna1` and `satimage1` datasets, whereas there is no significant difference between them on `iris1` and `ionosphere1` problems. The `iris1` and `ionosphere1` datasets were evaluated using ten-fold cross-validation, and both of them have a small number of instances compared to `dna1` and `satimage1`, which have fixed partitions of training and test data.

The unpaired t-test cannot carry on the remaining datasets, since the means and standard deviations are not given in both classifiers. Hence, it is difficult to compare the performance between GP and non-GP classifiers. However, some of them can be tested by using the z-test (Sprinthall, 2003) under the assumption that the datasets have a normal distribution with the means and standard deviations provided in Table 4.2. For example, if the data of the `banana` dataset are assumed to have normal distribution with a mean, μ, of 10.4 and standard deviation, σ, of 0.5, then the results from the GPC can be evaluated with the z-test, in which the GPC does not significantly differ from the SVM on the `banana` dataset. Under similar circumstances, the performance of GP and non-GP classifiers do not reveal significant differences on `breast cancer`, `bupa1`, `segmentation`, `thyroid1` and `wine` datasets, except for the `flare-solar1`, `ringnorm` and `splice` datasets, in which the differences are significant. Apart from the above datasets, there is not much information for comparisons between GP and non-GP classifiers although non-GP classifiers appear to perform better than GP classifiers due to the results in Table 4.2. Another problem that makes it difficult to compare the results is the difference in the experimental settings. For instance, the performance of C4.5 on `thyroid3` cannot be compared with the performance of GP-VA on `thyroid4`,

TABLE 4.2: Performance comparison between Gaussian process and best non-Gaussian process classifiers in the literature. The associated reference numbers in columns 2 and 7 are shown in Table 4.3 and the full names of the methods in columns 3 and 6 are shown in Table 4.4.

Dataset	Error						1% data size
	GP Classifier			Best non-GP in literature			
	Ref.	Method	Result	Result	Method	Ref.	
banana	[1]	GPC	10.5	10.4 ± 0.5	SVM	[2]	49
breast cancer	[1]	SPGPC	28.1	25.6 ± 4.4	SVM	[2]	1
bupa1	[3]	GP-Bayesian	30.8 ± 2.7	29.6	SVM	[4]	2
bupa2				26.9 ± 8.7	Neural ensemble	[5]	1
crabs	[6]	GP-HMC	2.50	0	SIP-LP	[7]	2
dna1	[8]	VBGPS	13.3 ± 1.3	8.9 ± 0.8	PWCP	[8]	29
dna2				4.131	SVM	[9]	12
flare-solar1	[1]	SPGPC	33.8	32.3 ± 1.8	SVM	[2]	4
flare-solar2				16.5	DeEPs	[10]	2
glass	[6]	GP-MAP	23.3	21.5	SIP-LP	[7]	1
German	[1]	GPC	23.0	22.2	RVM	[11]	3
heart1	[1]	GPC	17.8	16.6	SVM	[1]	1
heart2				14.8	k-means based	[12]	1
iris1	[13]	Lapace,VB	3.33 ± 3.51	2.2 ± 3.1	Neural ensemble	[5]	1
iris2				0	MSVM	[14]	1
ionosphere1	[15]	GPC-EP	4.85 ± 1.56	4.29 ± 1.63	SVM-EP	[15]	1
ionosphere2				1.3	3-NN+simplex	[16]	2
segmentation	[1]	GPC	2.7	2.4 ± 0.5	DAB-SVM	[17]	11
isolet	[8]	SGPISS	3.52	3.27	ECC with ANN	[18]	16
MNIST1	[19]	IVM	1.54 ± 0.04				100
MNIST2				0.38	VSVM	[20]	100
pima diabetes	[21]	GP-EP	22.63	17.95	GDA-LS-SVM	[22]	1
protein fold	[13]	GP-VA	38	44	SVM	[23]	4
ringnorm	[1]	SPGPC	1.4	1.5 ± 0.1	KFD	[24]	70
satimage1	[8]	VBGPM	12.0 ± 0.4	10.9 ± 0.4	PWCP	[8]	55
satimage2				7.65	SVM	[9]	20
sonar1	[21]	GP-EP	13.85	11.14	SVM	[21]	1
sonar2	[25]	Mean-field	7.7	2.9	NN	[26]	2
splice	[1]	GPC	11.5	9.5 ± 0.7	Reg-AdaBoost	[24]	22
thyroid1	[1]	SPGPC	3.7	3.7 ± 2.1	DAB-SVM	[17]	1
thyroid2	[15]	GPC-EP	2.77 ± 1.47				1
thyroid3				0.3	C4.5	[4]	24
thyroid4	[8]	GP-VA	3.86 ± 2.04				29
titanic	[1]	GPC	22.1	21.70	SVM	[27]	21
twonorm	[1]	SPGPC	2.6	2.39	SVM	[27]	70
USPS	[28]	SGPC	4.83	2.0	Comb-Tangent	[29]	21
waveform1	[1]	SPGPC	9.9	10.0	MProtSVM	[30]	46
waveform2				14.67 ± 0.005	SVM	[31]	46
waveform3	[8]	VBGPS	15.6 ± 0.7				49
wine	[13]	GP-VB	2.22 ± 3.88	0.562	SVM	[9]	1
wisconsin1	[21]	GP-EP	3.21	1.47	LS-SVM	[32]	1
wisconsin2	[33]	GP-Laplace	2.97				3
wisconsin3				0.71	SVM ensemble	[34]	1

even though both datasets are the same but have different training-test partitions. Consequently, the case cannot be compared and evaluated by the hypothesis test as the previous one.

With the four datasets on which the t-test is run, the corresponding p values (p_i) for each test are calculated and Fisher's method (Fisher, 1948) can be used by combining the p values from each test, and it is calculated by $-2\sum_{i=1}^{k} \ln(p_i)$, where k is the number of tests. The calculated value is tested against a χ^2 distribution of $2k$ degrees of freedom. Fisher's method shows that GP classifiers perform differently from non-GP classifiers at

TABLE 4.3: References of methods implemented in Table 4.2.

Ref.	reference	Ref.	reference
[1]	Naish-Guzman and Holden (2008)	[18]	Dietterich and Bakiri (1991)
[2]	Abe (2005)	[19]	Lawrence et al. (2003)
[3]	Gestel et al. (2002)	[20]	Dong (2003)
[4]	Gestel et al. (2004)	[21]	Kuss and Rasmussen (2006)
[5]	Zhou and Jiang (2004)	[22]	Polat and S. Güneş and A. Arslan (2008)
[6]	Barber and Williams (1997)	[23]	Ding and Dubchak (2001)
[7]	Figueiredo (2003)	[24]	Mika et al. (1999)
[8]	Girolami and Rogers (2006)	[25]	Opper and Winther (2000)
[9]	Hsu and Lin (2002)	[26]	Duch (2002)
[10]	Li et al. (2004)	[27]	Guo et al. (2005)
[11]	Tipping (2001)	[28]	Csató and Opper (2001)
[12]	Bagirov et al. (2003)	[29]	Keysers et al. (2002)
[13]	Girolami and Zhong (2006)	[30]	Aiolli and Sperduti (2005)
[14]	Liu et al. (2003)	[31]	Moguerza and Muñoz (2006)
[15]	Kim and Ghahramani (2006)	[32]	Polat and Güneş (2007)
[16]	Watkins and Boggess (2002)	[33]	Seeger (2000)
[17]	Li et al. (2005)	[34]	Sewak et al. (2007)

TABLE 4.4: Abbreviations and associated full names used in Table 4.2.

Abbreviation	Full name
ANN	artificial neural network
Comb-Tangent	combination of tangent vector and local representation
ECC	error correcting code
DAB-SVM	diverse AdaBoost SVM
De-EPs	decision-making by emerging patterns
EP	expectation propagation
GPC	Gaussian process classification
GDA-LS-SVM	generalized discriminant analysis and least square SVM
HMC	hybird Monte Carlo
IVM	informative vector machine
KFD	kernel Fisher discriminant
LA	Laplacian approximation
LS-SVM	least square SVM
MAP	maximum a posterior
MProtSVM	multi-prototype SVM
MSVM	multistage SVM
NN	neural network
PWCP	one-versus-one SVM with probabilistic outputs employing pairwise coupling
Reg-AdaBoost	regularized AdaBoost
RVM	relevance vector machine
SIP-LP	sparseness-inducing prior related to the Laplacian prior
SGPISS	sparse GP under informative sampling strategy
SGPC	sparse GP
SPGPC	sparse pseudo-input GPC
VA	variational approximation
VB	variational Bayes
VBGPM	variational Bayes GP with multiple length scale
VBGPS	variational Bayes GP with single length scale
VSVM	virtual SVM

the significance level of 0.05. Even such a naive combination of individual tests has severe limitations because the individual p value of the four tests are very different: $1.89\text{x}10^{-15}$, 0.455, 0.443 and $1.43\text{x}10^{-10}$. Thus, the information available in the published literature does not provide adequate information to make meaningful comparisons. It appears that the literature results make it difficult to conduct a comparison due to a lack of information and an invalid assumption of the t-test. It is evident that most of the best non-GPs are based on the SVM and their results are slightly better than the GP results. However, there is not enough information to form any conclusions.

The comparison of the results from the literature in this section shows that there is a lack of uncertainty in the results which will make it difficult for an appropriate comparison. Without considering uncertainty, the simple test could mislead one to a different conclusion due to the fact that the average accuracy itself also has the variability. As illustrated in Figure 3.1, the overlapping area implies the variability of the average accuracy. The larger the overlapping area, the more influence of the uncertainty, as the overlapping implies the possible variable space of the average accuracy. Therefore, considering only either the rank in the non-parametric test or the different value of the means in the parametric test, without taking care of the uncertainty, may not be appropriate in classification when there is information of both average accuracy and standard deviation from the cross-validation setting. From this viewpoint, we propose the hypothesis testing which considers both average accuracy and uncertainty from the cross-validation in this thesis. A possible approach to take the uncertainty into account is sampling a large number of samples from the performance distribution approximated from the cross-validation; then, based on these samples one is able to compute the p value for each sample and later make use of the p value to reject or not reject the null hypothesis. More details of this idea will be discussed in Chapter 5, in which we propose the hypothesis testing with uncertainty and investigate three sampling-based techniques.

4.2 Sparse Gaussian Discussion

Even though sparse Gaussian process algorithms in the literature have been proven to work as well as or sometimes better than using all of the data, it turns out that these demonstrations have all been conducted on small-scale problems. The largest problem to which sparse GP classification has been applied is the `MNIST` problem, with $60,000$ data points in Lawrence et al. (2003). In this study, while the authors' implementation of SVM and IVM achieve similar performance, the best quoted result in the literature has a lower error rate for SVM. Noting the comparison in Lawrence et al. (2003) with the subsampled images, one might suspect that the reduction in data quality may be the reason for similarity in the performances from IVM and SVM. Running IVM on the `MNIST` dataset at a higher image quality is the obvious next step that the authors have failed to perform. For lower data quality, we would expect the different methods

to perform comparably, and based on these results, one should not conclude that GP offers a competitive performance.

In Naish-Guzman and Holden (2008), a detailed comparison is presented, comparing the full Gaussian process classification, IVM, SVM and a novel sparse GP. In 9 of the 15 datasets of the conducted comparisons, the approximate GP method outperforms the full GP. In 6 datasets, the sparse approximation reduces to *just two* basis functions. In standard Bayesian pattern recognition, it is known that the optimal classifier for a two-class problem in which the two classes are Gaussian distributed with distinct means and covariance matrices is a quadratic classifier. While in the GP classifier the Gaussian basis functions are not intended to model the class conditional densities, this observation suggests that the benchmark problems considered are too simplistic and motivated for comparing the performance with the quadratic classifier for these six problems.

Table 4.5 compares the error rate between the SPGPC and quadratic classifiers in the six problems used in Naish-Guzman and Holden (2008). Four variants of quadratic classifier were implemented with different settings of covariance matrices ($\mathbf{\Sigma}$): (a) distinct full covariance matrices, (b) distinct diagonal matrices, (c) distinct isotropic covariance matrices, and, finally (d) identical full covariance matrix estimated from the pooled data of both classes. These results show that quadratic classifiers perform comparably to SPGPC, at a computational complexity that is relatively negligible.

TABLE 4.5: Error rate comparison between SPGPC in Naish-Guzman and Holden (2008) and quadratic classifiers with four variants of covariance matrices ($\mathbf{\Sigma}$) on two-class tasks in which SPGPC shows a high performance with only two basis functions.

Dataset	SPGPC	Quadratic classifier			
		$\mathbf{\Sigma}_1 \neq \mathbf{\Sigma}_2$	$\mathbf{\Sigma}_1 = \text{diag}(\mathbf{\Sigma}_1)$ $\mathbf{\Sigma}_2 = \text{diag}(\mathbf{\Sigma}_2)$	$\mathbf{\Sigma}_1 = \sigma_1 \boldsymbol{I}$, $\mathbf{\Sigma}_2 = \sigma_2 \boldsymbol{I}$	$\mathbf{\Sigma}_{\text{pooled}}$
breastcancer	0.28	0.31 ± 0.05	0.29 ± 0.05	0.30 ± 0.05	0.32 ± 0.04
diabetes	0.23	0.27 ± 0.02	0.26 ± 0.02	0.27 ± 0.0201	0.25 ± 0.02
heart	0.17	0.20 ± 0.03	0.16 ± 0.04	0.17 ± 0.038	0.16 ± 0.029
ringnorm	0.01	0.03 ± 0.003	$0.01 \pm 7.39e^{-4}$	$0.01 \pm 7.72e^{-4}$	0.25 ± 0.007
titanic	0.23	0.24 ± 0.06	0.26 ± 0.06	0.25 ± 0.006	0.231 ± 0.014
twonorm	0.03	0.03 ± 0.003	0.02 ± 0.001	0.02 ± 0.001	0.03 ± 0.002

The only problem that a GP classifier is reported to have a significantly higher accuracy rate than any other method is the protein structure classification problem considered in Girolami and Zhong (2006). Intrigued by this, we applied a multi-layer perceptron classifier to this problem. The network we implemented included 60 hidden nodes with softmax activation functions and used a conjugate gradient optimisation scheme. The neural network produced a 45.45 percent error rate, which is similar to the SVM result. An arguable question is: "Does the striking superiority of GP to the MLP and SVM performance seen here suggest a particular advantage for the GP classification on large

multi-class problems?". The available evidence is not strong enough to make this conclusion. This particular dataset has a fixed partition in training and test sets, and when only the result quoted on the test set is provided, one cannot be certain that it is not merely a fortuitous accident.

The meta-analysis of published work on Gaussian process classification suggests that experimental demonstrations of this technique have been conducted on relatively simple problems. While on simple problems the results are competitive with the best quoted results in the literature, the computational issues associated with standard GP methods prohibit their use on larger tasks. Sparse methods advanced to offset such difficulties have not been demonstrated on large enough tasks to be convincing. Another issue concerns predictive densities coming from a Bayesian treatment. This is a significant feature of GPs, yet there are no studies demonstrating their usefulness. Classification problems in benchmark datasets do not have labelling uncertainties associated with them. Furthermore, where a method systematically makes low confidence predictions, it is not possible to revisit the data source and shows that these are outliers in the data.

On the contrary, the SVM itself can also be considered a sparse method, since the aim of the SVM is to construct an optimal hyperplane from a small subset of data known as support vectors whose role is similar to the active set of sparse GP. To locate the support vectors, however, the SVM needs $O(N^3)$ for solving quadratic programming. Many researchers such as Cervantes et al. (2008), Ou et al. (2006) and Shin and Cho (2002) have attempted to preselect the data, expecting that these data are the candidate support vectors to alleviate the burden of solving the quadratic programming. We propose a number of data selection methods, with an aim to determine the candidate support vectors in order to enhance training time and performance. The proposed methods are presented in Chapter 6.

4.3 Summary

In this chapter, we carried out a meta-analysis to evaluate whether from the available published evidence one could reach some meaningful conclusion about the relative merits of Gaussian process classifiers with respect to the collection of other high-performing classifiers available in the literature. Except in a very small number of cases, we found that not only do the authors who advance new methods restrict the experimental demonstrations to small tasks, but they also fail to provide even the basic minimum information that would enable anyone to reach a meaningful conclusion about the methods. In order to conduct a proper comparison, we developed a systematic experimental setting which we present in the next chapter. The novelty of our contribution is the inclusion of uncertainties in the results that arise from random different partitionings of a dataset into training and evaluation sets.

Chapter 5

Testing for Significance with Uncertainty

In classification problems, there are inconsistencies when making comparisons because different papers evaluate their results in different ways. Many researchers have employed statistical tests to make such comparisons hoping that an algorithm actually produces results significantly differrent from others. Even though there are many statistical tests for making comparisons, one might overlook the assumptions before using them. There are two kinds of statistical tests: parametric and non-parametric tests as described in Chapter 3. An example of the parametric test is the paired t-test, which is a general comparison tool and used in the classification. Nonetheless, its assumptions of the homogeneity of variance and Gaussian distribution are usually ignored. Although the non-parametric tests do not require such assumptions, they have less power to detect the significant differrence. In the literature, many researchers reported their performances by using the average accuracy, the best accuracy ignoring the lower ones, or both mean and standard deviation. Salzberg (1997) recommends the ten-fold cross-validation (10CV) approach as a good choice for the trade-off between bias and variance rather than using a fixed training-test data partition. An additional by-product of doing this is the uncertainty or standard deviation obtained by the 10CV. However, many researchers report only the average mean without the uncertainty. Even though some of them provide the average accuracy and uncertainty, the statistical test that they employ only utilises the average score and ignores the uncertainty. In this chapter we propose a new approach to the testing hypothesis that considers the uncertainty. The main point of the method comes from the concept of sampling theory and the law of large numbers, which expects that the mean of a large number of repeatedly sampled data tends to be the expected value of the population. The first idea is based on Monte Carlo sampling, while the second idea is based on Markov chain Monte Carlo (MCMC) with a kernel density estimation (KDE). Given that one disadvantage of the Markov chain Monte Carlo method

is that it needs to be run for the long chain until convergence, a simple sampling method is proposed based on bootstrapping.

5.1 Hypothesis Testing with Uncertainty

While Salzberg (1997) recommends 10CV to compare classifiers and Demšar (2006) suggests using non-parametric tests for statistical comparisons of classifiers, the statistical tests are generally applied to the average accuracy from 10CV, as the performance metric assumes that the average accuracy is a good representative and ignores the uncertainty of the average accuracy. This section provides the details of the three proposed methods utilising the uncertainty together with the average accuracy for comparing classifiers.

5.1.1 Monte Carlo sampling

The idea of Monte Carlo sampling comes from the concept of Monte Carlo integration which randomly draws a large number $x_1, x_2, \ldots, x_n$ of random variables from a probability density function $p(x)$, and averages the function $f(x)$ respected to the distribution $p(x)$ as shown in the equation (5.1):

$$\int_a^b f(x)p(x)\,dx = E_{p(x)}[f(x)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i) \tag{5.1}$$

In our situation, we could imagine the function is the difference between the two accuracy rates randomly drawn from the Gaussian distributions of corresponding classifiers' performance obtained from the 10CV. For instance, we use SVM and GP, on M datasets. The performances of these classifiers are $A_{SVM,m} \pm V_{SVM,m}$ and $A_{GP,m} \pm V_{GP,m}$, where A stands for average accuracy and V denotes standard deviation across the 10CV on the mth dataset. We assume these numbers are Gaussian distributed and the performance are sampled from these Gaussians.The differences in performance $d_m = A_{SVM,m} - A_{GP,m}$ for the M datasets, $m = 1, \ldots, M$ are calculated. Based on those d_m we calculate a p value. This process is repeated several times. By repeating the process a large number of times, e.g., 10000 iterations, we have the distribution of the p values from the process, and the median of the p values is calculated to make the decision whether to reject the null hypothesis. The reason we use the median of the p values as the threshold against the significant level is because the distribution of those p values is not normally distributed; hence, the mean value should not be the representative of the centre of the distribution. In contrast, the median is the better representative as it is the centre value between the mean and mode in the skewed distribution. In addition, the average performance rank of the two classifiers, if they perform comparably, should be located somewhere around the middle of the rank; that is, the median. This is also a concept of

comparison in the non-parametric test, which compares the median rather than mean as employed in the parametric test. Moreover, the median is more robust to noise or outlier than mean. The Monte Carlo sampling approach is shown in Algorithm 3:

Algorithm 3 : Monte Carlo Sampling approach

1: Sample $A_{SVM,m}$ and $A_{GP,m}$ from the distributions $\mathcal{N}(A_{SVM,m}, V_{SVM,m})$ and $\mathcal{N}(A_{GP,m}, V_{GP,m})$, for each dataset $m = 1, \ldots, M$.
2: Compute $d_m, m = 1, \ldots, M$.
3: Compute one-tailed p value of significance.
4: Repeat step 1-3 many times to get a distribution of those p values, at which the null hypothesis may be accepted or rejected.
5: Make a final accept/reject decision based on the median of p values.

In order to verify the effectiveness of the proposed sampling-based hypothesis testing, we conducted an experiment using two class synthetic data in which the ground truth is known and generated by two normal densities. The normal densities are of different means ($\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2$), but have the same spherical covariance matrices ($\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \sigma^2 \boldsymbol{I}$). The means were randomised and the covariance matrices were set with a value of two on the diagonal, and zero elsewhere. There are three dataset types: three datasets with two features, three datasets with three features, and four datasets with ten features, and each class has 300, 400, and 500 data on the two, three, and ten-feature problems, respectively. Under this circumstance we know the ground truth that quadratic discriminant analysis (QDA) behaves similarly to linear discriminant analysis (LDA) (Duda et al., 2001, Chapter 2); hence, both classifiers should perform equivalently. With this ground truth prior to the realisation, these experiment settings will also be used to verify the MCMC with KDE and the bootstrapping approaches, as well as the Monte Carlo sampling in this section. Table 5.1 shows the performances of QDA and LDA on these synthetic datasets. In this experiment, the Wilcoxon matched-pairs signed-ranks test rejects the null hypothesis as $|z| = 2.80$, which is greater than 1.65; in other words, its p value is 0.0026, whereas the Monte Carlo sampling approach accepts the null hypothesis since the median p value is 0.19, which is greater than the 0.05 significance level. Therefore, the Monte Carlo sampling, which makes use of both average accuracy and standard deviation from the 10CV, gives the consistent conclusion with the ground truth, but the Wilcoxon matched-pairs signed-ranks test fails to detect the ground truth as a result of using average accuracy alone.

5.1.2 MCMC with kernel density estimation

The Monte Carlo sampling method simply assumes that the performance for each dataset by each classifier is Gaussian distribution, although this may not be true for every dataset. However, the distribution could be estimated from the data by using a kernel density estimation in which the Parzen window (Parzen, 1962) is used. The distribution

TABLE 5.1: Summary results obtained by LDA and QDA applied on ten binary class synthetic datasets.

dataset	LDA	QDA
1	71.50 ± 5.12	72.33 ± 5.73
2	71.33 ± 5.49	73.17 ± 7.26
3	93.33 ± 3.04	93.67 ± 3.50
4	80.13 ± 4.06	81.63 ± 3.59
5	91.88 ± 3.50	92.00 ± 4.46
6	77.13 ± 5.30	78.63 ± 3.41
7	97.70 ± 2.11	98.40 ± 1.26
8	98.50 ± 1.08	99.20 ± 0.92
9	97.40 ± 1.58	98.30 ± 1.64
10	98.10 ± 1.97	99.60 ± 0.70

of performance on dataset m could be approximated by

$$f_m(x) = \frac{1}{10h} \sum_{i=1}^{10} K(\frac{x - x_i}{h}) \tag{5.2}$$

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp\{-0.5x^2\} \tag{5.3}$$

where h is the smoothing parameter determined by cross-validation and x_i is the accuracy of the ith fold of the 10CV on dataset m. We draw a large number of samples similar to the first approach, but the samples are drawn by MCMC in which the target distribution is estimated by the KDE. The basic idea behind the KDE relies on the fact that the probability P that a vector falls in a very small region R, in which the density $p(\boldsymbol{x})$ does not vary much, is given by:

$$P = \int_R p(\boldsymbol{x})\, d\boldsymbol{x} \approx p(\boldsymbol{x}) \int_R d\boldsymbol{x} = p(\boldsymbol{x})V \tag{5.4}$$

where V is the volume of R. Let h be the length of an edge of the hypercube R; hence, $V = h, h^2$ and h^3 for 1D, 2D and 3D, respectively. If we have drawn n samples according to the density $p(\boldsymbol{x})$, and there are k out of n samples falling in R, the proability P is $\frac{k}{n}$. As a result, the density $p(\boldsymbol{x}) = \frac{k}{nV}$. The total number k samples falling in R can be calculated by using a window function $\sum_{i=1}^{n} K(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h})$. In our density approximation $f(x)$ for 1D, we use the window function as shown in equation (5.3) and this leads to the density estimation in equation (5.2) where we have $n = 10$ from the 10CV.

Then the MCMC is employed to draw many samples. The concept of MCMC comes from the Markov process, that is, the future state depends only on the present state. Let X_t be the value of a random variable at time t. The Markov process implies $P(X_{t+1}|X_0, X_1, \ldots, X_t) = P(X_{t+1}|X_t)$, and a Markov chain is a sequenece $(X_0, X_1, \ldots, X_n)$ generated by the Markov process. The probability of moving from state i to state

j in a single step is defined by the probability transition matrix $\boldsymbol{P}$ in which $p_{i,j}$ is $P(X_{t+1} = s_j | X_t = s_i)$, whereas the probability of moving from state i to state j in n step is $p_{i,j}^{(n)} = P(X_{t+n} = s_j | X_t = s_i)$, which is the corresponding ij-th element of $\boldsymbol{P}^n$. Let $\pi_i(t) = P(X_t = s_i)$ be the probability that the chain is in state i at time t and $\boldsymbol{\pi}(t)$ denote a row vector of the state space probabilities at time t. By using Chapman-Kolmogorov equation (Kolmogorov, 1931), we obtain $\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0)\boldsymbol{P}^t$, where all elements of $\boldsymbol{\pi}(0)$ are zero except for a single element of 1, corresponding to the starting state of the chain. We need the Markov chain to reach a stationary distribution $\boldsymbol{\pi}^*$, and a sufficient condition for the stationary distribution is that the detailed balance equation holds:

$$p_{ij}\pi_i^* = p_{ji}\pi_j^* \tag{5.5}$$

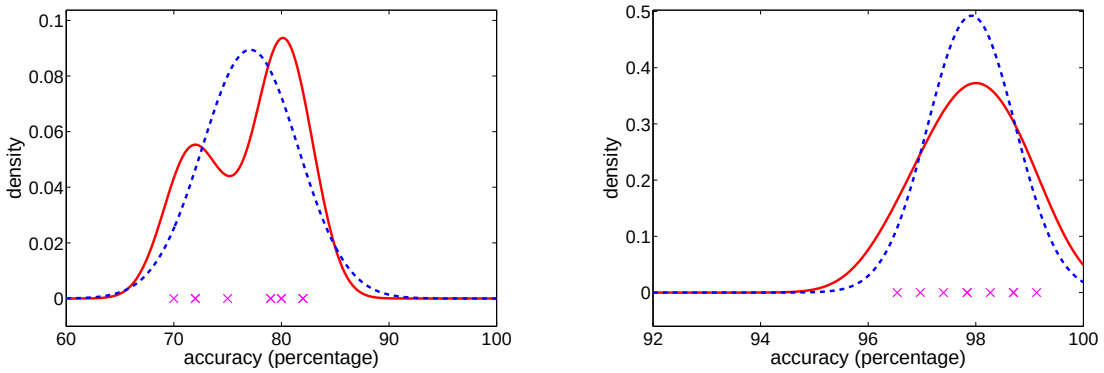
The Markov chain that satisfies the detailed balance is said to be reversible. In our approach, we use the Metropolis-Hastings algorithm (Metropolis and Ulam, 1949; Hastings, 1970) to draw the samples from the density distribution $f(x)$ estimated in the KDE step. It uses a proposal density $q(x'; x^t)$, which depends on the current state x^t, to generate a new sample x'. The new sample is accepted as the next state $x^{t+1} = x'$ if α drawn from a standard uniform distribution $U(0, 1)$ satisfies:

$$\alpha < min\left\{\frac{f(x')\, q(x^t; x')}{f(x^t)\, q(x'; x^t)}, 1\right\} \tag{5.6}$$

If the new sample is not accepted, there is no move to the new state, that is, $x^{t+1} = x^t$. In our implementation, a Gaussian function $\mathcal{N}(x^t, \sigma^2)$ is used for the proposal $q(x'; x^t)$, where σ is set to 10. Following a sufficient burn-in period, which is set to 10000 iterations, the chain approaches the stationary distribution, and 10000 samples after the burn-in iterations are used as the samples from our $f(x)$.

Figure 5.1 illustrates two examples in which the approximation of the performance distribution may or may not agree with the Gaussian distribution. It is evident that the sample points indicating the accuracies from the 10CV shown as crosses in Figure 5.1(a) are not properly approximately sampled by unimodal distribution compared to the bimodal approximation by KDE. On the other hand, Figure 5.1(b) shows another case in which KDE captures the unimodal distribution as the Gaussian distribution. In the latter case, the Gaussian assumption may be acceptable; nonetheless, the Gaussian assumption in the Monte Carlo sampling approach is not likely to be true in many cases. In the second approach, we draw samples from the estimated distribution by the Markov chain Monte Carlo and then we follow the similar steps described in Algorithm 4. We use the median of the p values from those samples as the same reason explained in the Monte Carlo sampling approach to reject or not reject the null hypothesis.

The same experiment with the ground truth of LDA and QDA settings as explained in the previous subsection and the results in Table 5.1 are evaluated by MCMC with



(a) (b)

FIGURE 5.1: Comparison between the Gaussian assumption (dashed line) and kernel density estimation, KDE (solid line), in which (a) KDE captures the bimodal distribution on the German dataset and (b) both distributions are unimodal on the image dataset regarding the sample data points (crosses) from 10CV.

Algorithm 4 : MCMC with kernel density estimation approach

1: Estimate the distributions by Parzen window.
2: Sample a pair of value, $A_{SVM,m}$ and $A_{GP,m}$, from the estimated distributions by MCMC.
3: Compute $d_m, m = 1, \ldots, M$.
4: Compute one-tailed p value of significance.
5: Repeat step 2-4 many times to get a distribution of those p values, at which the null hypothesis may be accepted or rejected.
6: Make a final accept/reject decision based on the median of the p values.

KDE. This approach gives the median p value 0.21, which is greater than the significance level, so the null hypothesis cannot be rejected. It shows the same conclusion as the first approach and consistency with the ground truth that is already known, whereas the non-parametric test rejects the null hypothesis. The evaluation shows that taking uncertainty into account would be more appropriate for the hypothesis testing in classification.

5.1.3 Bootstrapping

Another way to circumvent the Gaussian assumption is to apply the bootstrapping technique, which is a class of the resampling method; it is a non-parametric method and can be used for estimating the sampling distribution of a statistic. Figure 5.2 shows the concept of the bootstrap, in which a data is randomly drawn from the original data in each round and put back to the original data before beginning the next round of the sampling. This process of sampling is referred to as resampling with replacement. It is possible that the same data is drawn next time, as illustrated in the figure, and the last approach is based on this scheme.

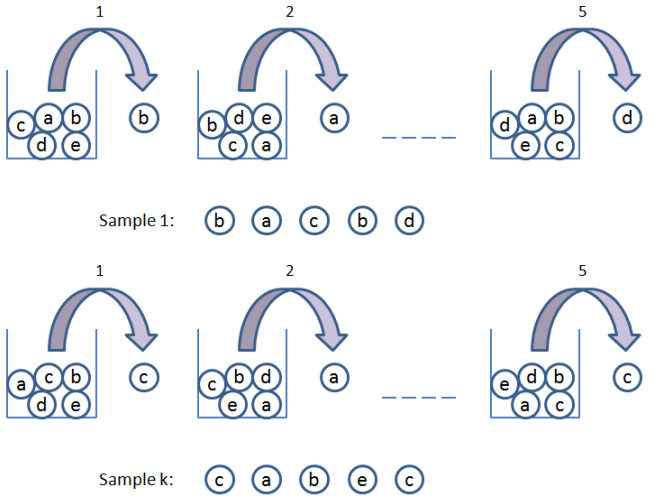


FIGURE 5.2: Illustration of bootstrapping resampling technique. In this example, each bucket has five data, and the arrow denotes randomly resampling one data from the bucket. The first sample from the bootstrap is b-a-c-b-d; this example repeats k times of the bootstrap, and the kth sample is c-a-b-e-c. Data drawn out from the bucket will be taken to the bucket before the resampling proceeds.

In our approach, since the experiments are run on the 10CV, each dataset has ten accuracy values from these ten folds. The accuracy of each dataset is sampled with replacement from those ten folds. The same steps as the Monte Carlo sampling approach are followed, except that the pair of numbers in Step 1 is obtained by random sampling with replacement from those ten accuracy values given by the associated classifiers for each dataset. This process is completed many times; for example, we run 10000 iterations in the implementation, and a judgment is made based on the median of the p values with the same reason in the first approach. Algorithm 5 shows the pseudocode of the bootstrapping approach.

Algorithm 5 : Bootstrapping approach

1: Random Sampling with replacement from 10CV on each dataset
2: Compute $d_m, m = 1, \ldots, M$.
3: Compute one-tailed p value of significance.
4: Repeat step1-3 many times to get a distribution of those p values, at which the null hypothesis may be accepted or rejected.
5: Make a final accept/reject decision based on the median of the p values.

The same synthetic data and experimental settings in the previous two subsections are evaluated by the bootstrapping approach. The median p value by the approach is 0.19, which is greater than the significance level, that is, we cannot reject the null hypothesis. The conclusion conforms with the ground truth that LDA and QDA perform comparably under the experimental settings. Table 5.2 summarises the p values from the proposed sampling-based approaches. It shows that the proposed approaches have the

same conclusion, which agrees with the ground truth, while the Wilcoxon matched-pairs signed-ranks test fails to detect the ground truth due to that fact that it considers only average accuracy for making the hypothesis test. The result on the synthetic data verifies that the hypothesis testing with considering the uncertainty would be more appropriate than the test ignoring the uncertainty. The real datasets are also used to evaluate the effectiveness of the proposed sampling-based hypothesis testing in next section.

TABLE 5.2: Summary results on synthetic data evaluated by the proposed sampling-based approaches and Wilcoxon matched-pairs signed-ranks test (WMPSR).

Approach	p value	H_0: LDA = QDA
WMPSR	0.0026	reject
Monte Carlo sampling	0.19	not reject
MCMC with KDE	0.21	not reject
Bootstrapping	0.19	not reject

5.2 Experimental Settings and Results

To determine the effectiveness of the above proposed statistical tests, which consider the uncertainty from the 10CV, the experiments were carried out on a number of real datasets and discussed in the subsequent sections.

5.2.1 Data and implementations

Table 5.3 shows the list of datasets and their characteristics used in the experiments. Most of the datasets were selected from UCI (Asuncion and Newman, 2007) and FIRST IDA[1] repositories because they are publicly available and commonly used as benchmark datasets. Chemoinformatics (`chemo`), protein (`SCOP`) and gene expression (`lung cancer`) are the datasets used in Rhodes et al. (2000), Ding and Dubchak (2001) and Landi et al. (2008), respectively. These datasets were added to the experiments in order to diversify extreme cases; for instance, `lung cancer` has a very high dimensionality with a small number of data, and `SCOP` has 27 classes for six parameter datasets from which two datasets were selected. Fifteen datasets are binary problems, while twelve datasets are multi-class problems. Figure 5.3 illustrates an overview of the datasets in 3D in which the x-axis, y-axis and z-axis represent the number of classes, number of dimensions and number of data, respectively. The datasets vary from 2 to 27 classes and the dimensions vary from 2 to 22283. Given that highly expensive time computation ($\mathcal{O}(N^3)$), in which N is the number of training data, the size of each dataset is restricted to less than ten thousand instances; for example, the subset of data in `MNIST`[2] were randomly selected

[1] Available from `http://ida.first.fhg.de/projects/bench/benchmarks.htm`
[2] Available from `http://yann.lecun.com/exdb/mnist/`

from each class, preserving the same ratio of the number of training data to the number of test data; except for `shuttle`, the largest data size is 58000 as an extreme case.

TABLE 5.3: Summary of the datasets used in the experiments.

dataset	#class	#dim	#data		dataset	#class	#dim	#data
banana	2	2	5300		satimage	6	36	6435
breast cancer	2	9	277		SCOP-CSHPVZ	27	125	698
chemo	2	992	5747		SCOP-CSH	27	62	698
diabetes	2	8	768		segment	7	19	2310
flare-solar	2	9	1066		shuttle	7	9	58000
German	2	20	1000		sonar	2	60	208
glass	6	9	214		splice	2	60	3175
heart	2	13	270		thyroid	3	5	215
image	2	18	2310		titanic	2	3	2201
ionosphere	2	34	351		twonorm	2	20	7400
iris	3	4	150		vowel	11	10	990
lung cancer	2	22283	107		waveform	3	21	5000
MNIST	10	784	3500		wine	3	13	178
ringnorm	2	20	7400					

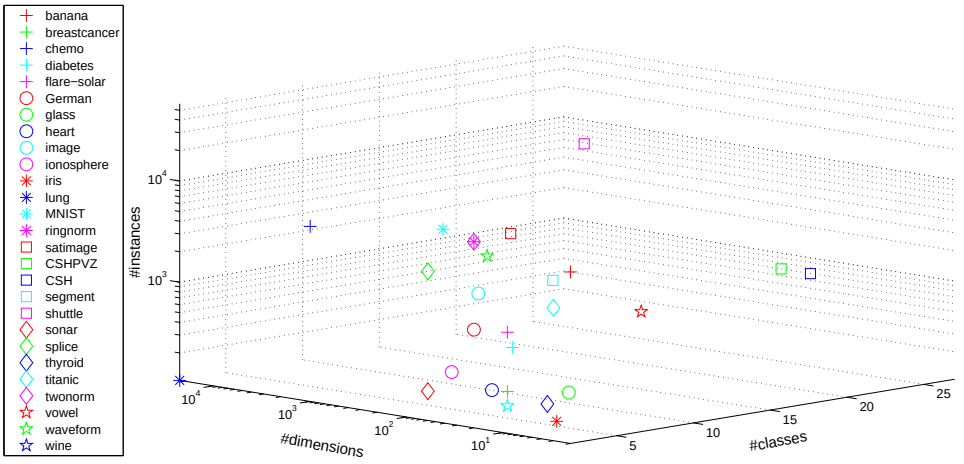


FIGURE 5.3: An overview of all datasets in 3D.

All datasets were pre-processed using normalisation or scaling. Datasets from FIRST IDA, `banana`, `breast cancer`, `flare-solar`, `German`, `image`, `ringnorm`, `splice`, `titanic`, and `twonorm`, had been normalised to have a zero mean and standard deviation of one in each feature. The remaining datasets were scaled to the range between -1 and 1 in each feature. Three classifiers: SVM, GP and FLDA, were chosen to run on the datasets with 10CV because of the intention to compare the proposed tests with the Friedman test which should be used when there are more than two classifiers. For SVM, the RBF kernel was applied to all datasets. The trade-off parameter (C) and the RBF kernel parameter (γ) were tuned in the range $C \in \{2^{-2}, 2^{-1}, 2^{0}, \ldots, 2^{12}\}$ and $\gamma \in \{2^{-10}, 2^{-9}, 2^{-8}, \ldots, 2^{4}\}$. For GP, the squared exponential covariance function with an isotropic distance measure and cumulative Gaussian likelihood function was applied to all datasets. Laplace's approximation was used for approximate inference. The logarithm

of the length-scale and signal variance parameters in the covariance function was tuned into the same range of $\{0, 5, 10, 15, 20\}$. Due to the expensive computation on large datasets, particularly in GP, the 10CV strategy from Hsu and Lin (2002) was followed and implemented for all datasets. A dataset is randomly separated into ten folds. Each time, nine folds are used as the training data and the remaining fold is used as the test data. Then all pre-defined parameter values are used for the training data and evaluated on the test data. The best average cross-validation rate is recorded. There are two versions of FLDA implemented in the experiments. As there is no kernel parameter in FLDA, the first version is performed as explained in Chapter 2 in Section 2.4, whereas $\boldsymbol{w}$ is computed from the entire data in the second version, thus offering a better cross-validation rate than the former version.

The aforementioned settings were used to verify the performance of the proposed tests compared with other standard statistical tests because the ground truth which reveals that at least a difference among the classifiers is already known. In a K-class problem, where $K > 2$, the One-versus-All scheme was implemented because it requires K binary classifiers rather than $\frac{K(K-1)}{2}$ classifiers, as required in One-Verus-One or Directed Acyclic Graph schemes. The software used in the experiments is SVM^{light} by Joachims (1999) for SVM and *GPML* by Rasmussen and Williams (2006) for GP.

5.2.2 Results and discussions

Table 5.4 shows the average accuracies and standard deviations of each classifier's performance on the datasets. We then evaluated the results using the Wilcoxon matched-pairs signed-ranks test, Friedman test and all of the proposed methods to determine whether there is any significant difference in performance among classifiers and also to compare the performance of these tests with the proposed methods. As shown in Chapter 4, the best quoted results in the literature on the datasets classified by the GP-based methods have a higher error rate than the best quoted results on the same datasets classified by SVM-based methods; however, these results were evaluated using different experimental settings. This is a good opportunity to compare the performance between SVM and GP under the same settings. It has been known that FLDA is a suitable classifier for the linear separable cases and works poorly on non-linear separable problems, whereas SVM and GP are capable of both linear and non-linear classification. Therefore, we pay more attention to SVM and GP than FLDA in the experiments. FLDAs were added to Table 5.4 as a result of multiple testing in the Friedman test.

Figure 5.4 is the scatter plot of the average accuracies classified by SVM and GP from Table 5.4. The diagonal line shows the case that SVM and GP give the same accuracies. Most of the points are above the diagonal line, which indicates higher accuracies by SVM. The uncertainty is embedded in the plot and represented by the length of the vertical and horizontal lines; the vertical length implies the SVM's variation and the

horizontal length indicates the GP's variation corresponding to the same dataset. The vertical lines are slightly shorter than the horizontal lines, which means that the SVM has a smaller variation than the GP.

TABLE 5.4: Average accuracy rate comparison between SVM and GP as well as Fisher's linear discriminant ($FLDA_1$ & $FLDA_2$ as described) classifiers.

dataset	SVM	GP	$FLDA_1$	$FLDA_2$
banana	90.70 ± 1.09	90.43 ± 1.17	55.77 ± 1.69	56.15 ± 1.88
breast cancer	75.14 ± 10.97	75.14 ± 9.14	73.76 ± 10.59	73.76 ± 11.24
chemo	96.17 ± 0.89	94.15 ± 1.33	93.88 ± 1.18	97.16 ± 0.68
diabetes	78.26 ± 2.24	77.61 ± 3.43	77.74 ± 3.06	78.39 ± 2.85
flare-solar	67.54 ± 2.20	67.18 ± 2.78	66.79 ± 2.39	66.98 ± 2.43
German	77.10 ± 4.46	76.40 ± 4.60	75.40 ± 5.44	76.00 ± 5.14
glass	73.70 ± 9.55	69.81 ± 9.73	61.24 ± 11.98	64.57 ± 9.39
heart	84.82 ± 7.08	84.07 ± 6.54	84.07 ± 7.21	84.81 ± 7.50
image	97.92 ± 0.81	97.88 ± 1.11	83.94 ± 1.45	84.24 ± 1.42
ionosphere	95.14 ± 4.05	90.87 ± 5.01	85.46 ± 5.48	90.00 ± 3.39
iris	97.33 ± 4.66	96.67 ± 6.48	83.33 ± 5.67	84.67 ± 7.06
lung cancer	90.91 ± 7.42	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
MNIST	96.23 ± 1.27	93.80 ± 1.71	78.89 ± 2.10	91.77 ± 1.41
ringnorm	98.58 ± 0.35	98.15 ± 0.42	77.07 ± 1.42	77.30 ± 1.47
satimage	92.62 ± 0.84	91.67 ± 0.86	75.67 ± 1.71	76.04 ± 1.71
SCOP-CSH	61.59 ± 7.22	57.73 ± 5.39	33.08 ± 9.00	41.41 ± 8.32
SCOP-CSHPVZ	62.75 ± 6.07	56.73 ± 4.31	36.42 ± 5.46	54.46 ± 4.60
segment	97.49 ± 1.43	97.66 ± 1.06	84.55 ± 2.32	84.81 ± 2.31
shuttle	99.92 ± 0.03	78.60 ± 0.47	87.24 ± 0.42	87.24 ± 0.42
sonar	88.42 ± 6.52	87.52 ± 9.01	75.36 ± 10.06	90.33 ± 6.49
splice	92.13 ± 1.40	91.28 ± 1.75	83.94 ± 1.54	85.32 ± 1.63
thyroid	97.73 ± 4.42	96.82 ± 5.69	86.28 ± 8.40	86.28 ± 8.40
titanic	79.06 ± 3.25	79.06 ± 3.25	77.60 ± 3.17	77.60 ± 3.17
twonorm	97.89 ± 0.55	97.80 ± 0.56	97.76 ± 0.60	97.89 ± 0.58
vowel	99.60 ± 0.52	98.79 ± 1.70	41.41 ± 5.53	46.77 ± 3.05
waveform	87.18 ± 1.85	86.80 ± 1.47	85.94 ± 2.01	86.52 ± 1.69
wine	99.44 ± 1.76	98.33 ± 2.68	98.89 ± 2.34	100.00 ± 0.00

In order to determine the effect of the data size on accuracy and uncertainty, we plot the distributions of accuracy and uncertainty by SVM and GP at the top and bottom of Figure 5.5, respectively. As expected, the larger size of the data, the lower the uncertainty. However, a larger data size does not guarantee higher accuracy than a smaller data size. Outliers could be a reason for the low average accuracy and high uncertainty, even though the dataset is large. In SVM, partial instances known as support vectors are important to constructing the decision boundary; therefore, the same results occur regardless of the data size, provided that there are the same support vectors. Figure 5.6 shows the graphs between the number of dimensions and average accuracy as well as between the number of dimensions and standard deviation by the SVM and GP classifiers on the datasets. It appears that the accuracies by SVM and

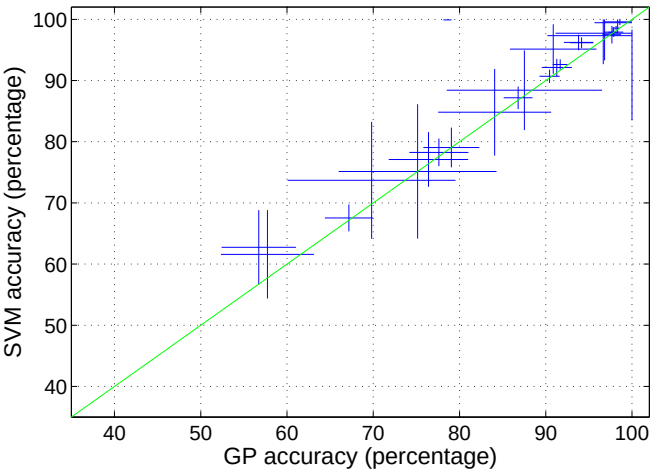


FIGURE 5.4: Scatter plot between SVM and GP accuracies. The crosspoints between horizontal and vertical are the mean accuracies. The uncertainties of SVM accuracies are presented by the vertical lines, whereas the uncertainties of GP accuracies are presented by the horizontal lines.

GP have a higher rate and the uncertainties by SVM and GP have a lower rate when the number of dimensions is greater than 100 dimensions while the effect of dimensions fluctuates at the lower number of dimensions. This observation suggests that larger dimensions could improve performance.

Table 5.4 clearly shows that the average accuracies by SVM are larger than the average accuracies by GP and FLDAs classifiers in most of the datasets. Some classifiers give the same average accuracies on the same datasets. Even though most of the average accuracies by SVM are slightly larger than the average accuracies by GP, there are some datasets in which the SVM performs significantly better than GP, for example, `shuttle`. There are some datasets in which the GP or FLDAs have higher accuracies than the SVM such as in `lung cancer`. On the other hand, the uncertainties by the classifiers appear to have similar trend in most of the datasets.

Non-parametric tests rather than parametric tests were run because the assumptions, e.g., homogeneity of variance, for the parametric tests may have been invalid, as the parameters of the population are not known in practice. Hence, non-parametric tests are safer than parametric tests (Demšar, 2006). For the Wilcoxon matched-pairs signed-ranks test, $|z| = 3.75, 4.25, 3.53, 3.85, 2.78, 4.25$ on the pairs of SVM and GP, SVM and FLDA_1, SVM and FLDA_2, GP and FLDA_1, GP and FLDA_2, FLDA_1 and FLDA_2, respectively. The absolute z values are greater than the one-tailed critical value $|z| = 1.65$. In other words, the p value is lower than the significance level at 0.05; hence, all null hypotheses are rejected.

The results of the Friedman test are shown in Table 5.6. The one-tailed critical difference value at the 0.05 significance level is 0.81, and the average ranks of SVM, GP, FLDA_1 and FLDA_2 are 1.37, 2.33, 3.67 and 2.63, respectively. The absolute differences between the

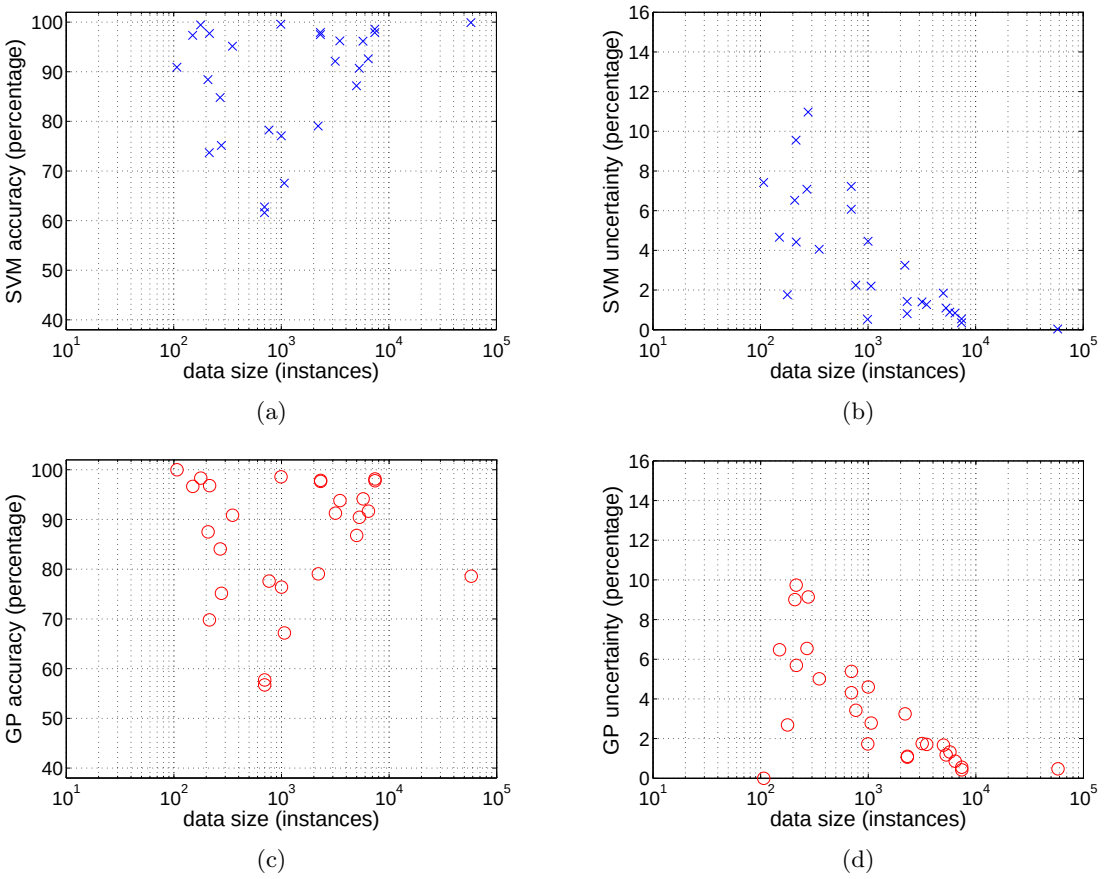


FIGURE 5.5: Distributions of accuracy and uncertainty against data size. The accuracy and uncertainty by SVM plotted on the y-axis against data size on the x-axis are shown in panel (a) and (b), while the similar plots by GP are shown in panel (c) and (d) respectively.

TABLE 5.5: Summary results of the Wilcoxon matched-pairs signed-ranks test on the datasets classified by SVM, GP, and FLDAs.

Avs$B$	$\|z\|$	p value	$H_0 : A = B$
SVM vs GP	3.75	8.84×10^{-5}	reject
SVM vs FLDA_1	4.25	1.07×10^{-5}	reject
SVM vs FLDA_2	3.53	2.08×10^{-4}	reject
GP vs FLDA_1	3.85	5.91×10^{-5}	reject
GP vs FLDA_2	2.78	0.0027	reject
FLDA_1 vs FLDA_2	4.25	1.07×10^{-5}	reject

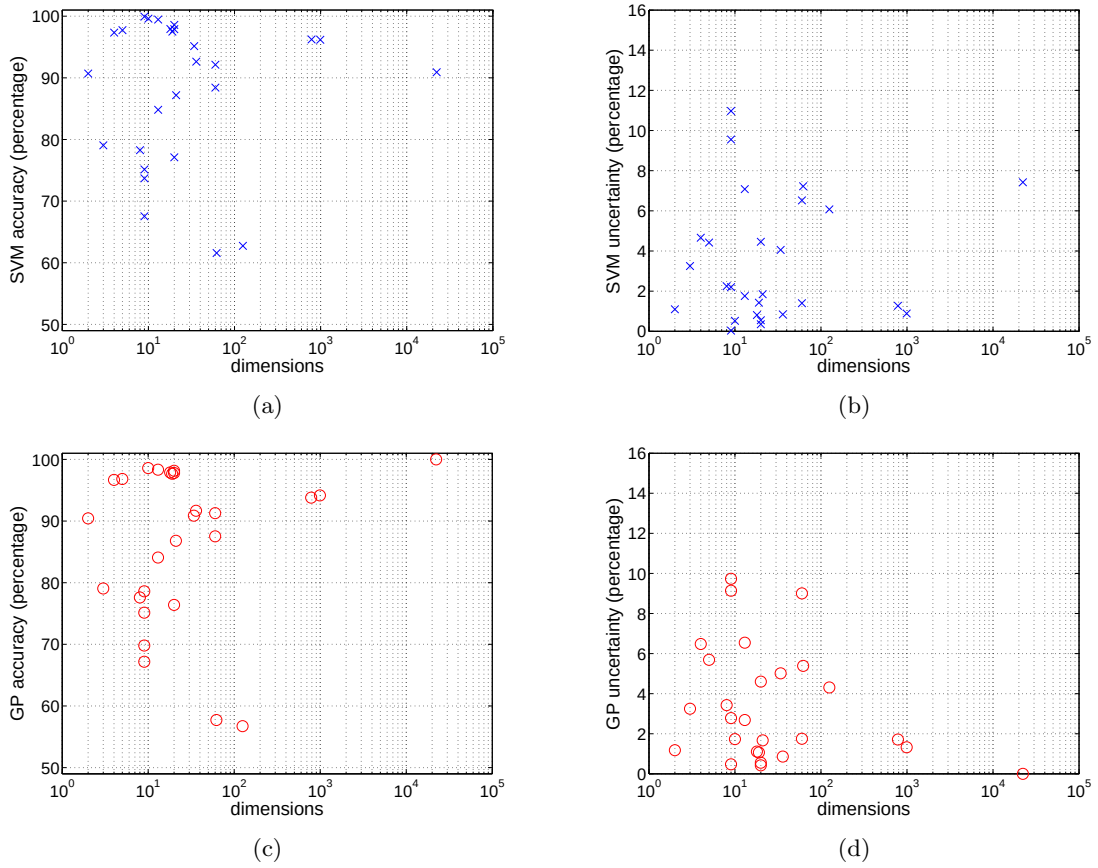


FIGURE 5.6: Distributions of accuracy and uncertainty against the number of dimensions. The accuracy and uncertainty by SVM plotted on the y-axis against the number of dimensions on the x-axis are shown in panel (a) and (b), while the similar plots by GP are shown in panel (c) and (d), respectively.

average rank pairs are $0.96, 2.30, 1.26, 1.34, 0.30, 1.04$ for the pairs of SVM and GP, SVM and FLDA_1, SVM and FLDA_2, GP and FLDA_1, GP and FLDA_2, FLDA_1 and FLDA_2, respectively. These numbers, except for the pair of GP and FLDA_2, are greater than the critical difference value 0.81, which means that the corresponding null hypothesis can be rejected. The null hypothesis between GP and FLDA_2 cannot be rejected, as the difference of the average ranks is smaller than the critical difference value. If SVM is used as the control group, the Friedman test shows that no classifier among three is statistically better than SVM. The Friedman test and the Wilcoxon matched-pairs signed-ranks test agree on the same conclusion, except that the former test cannot reject the null hypothesis on the pair of GP and FLDA_2, whereas the latter test rejects it.

Subsequently, we employed the proposed methods. Figure 5.7 shows the distribution of $\ln p$ values by the Monte Carlo sampling method, running 10000 iterations. All graphs are not normally distributed, even though some graphs are a similar shape as the skewed normal distribution. As a result, the median p value is used as an ad hoc threshold against the significance level. The corresponding median p values, depicted by dashed lines in the graphs, for the pairs of SVM and GP, SVM and FLDA_1, SVM and

TABLE 5.6: Average ranks and test statistic by performing the Friedman test on Table 5.4.

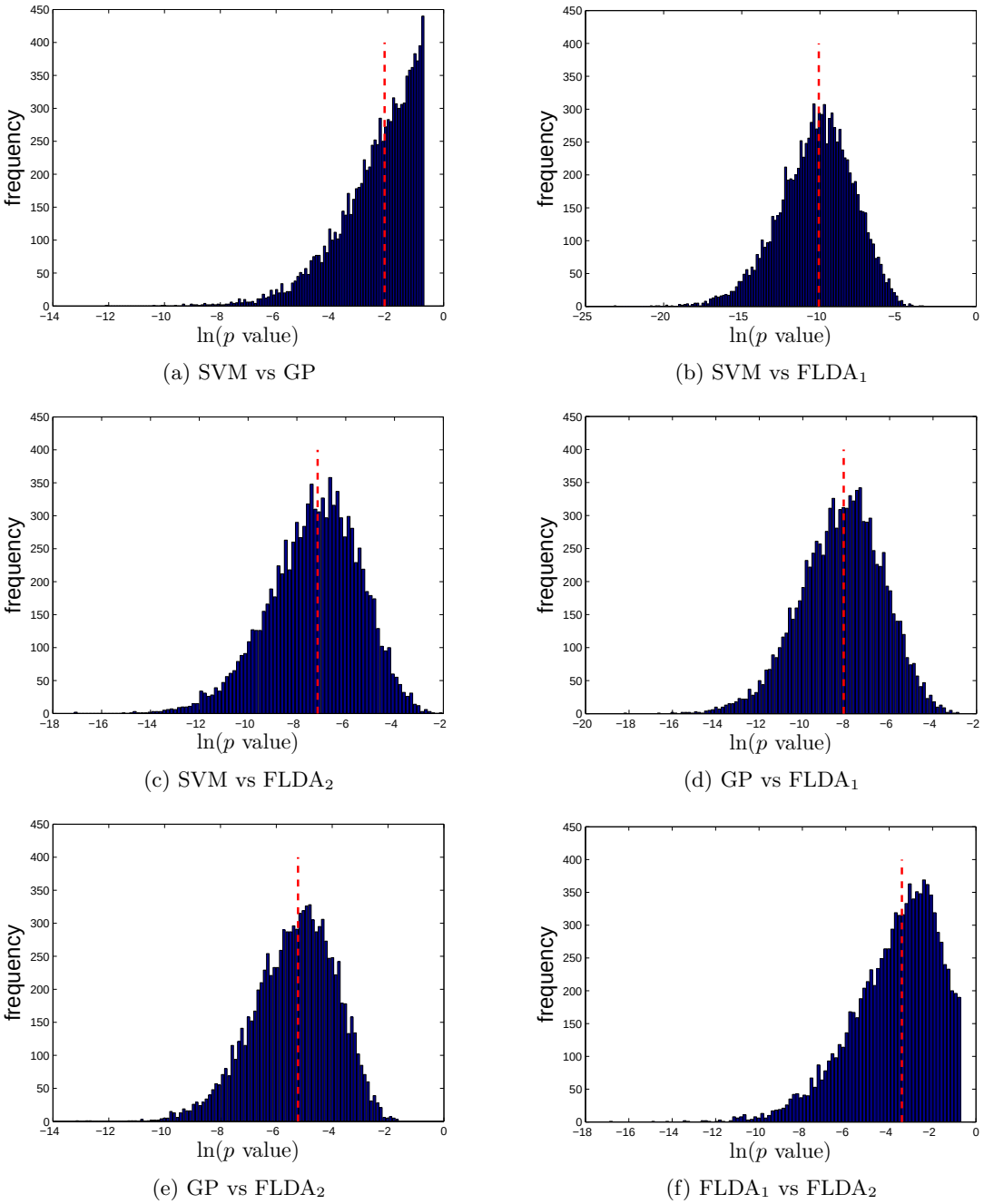
Rank			Test Statistic	
	Average Rank		Statistic	Value
SVM	1.37		F_F	30.08
GP	2.33		df_1	3
FLDA_1	3.67		df_2	78
FLDA_2	2.63		$CD = 2.291\sqrt{\frac{4(5)}{6(27)}}$	0.81

FLDA_2, GP and FLDA_1, GP and FLDA_2, and FLDA_1 and FLDA_2 are 0.12, 4.08×10^{-5}, 7.84×10^{-4}, 2.99×10^{-4}, 0.0054, and 0.033, respectively. The Monte Carlo sampling method cannot reject the null hypothesis of the SVM and GP pair because the median is greater than the significance level, whereas it rejects others. The bootstrapping method produces the corresponding median p values of 0.12, 3.33×10^{-5}, 6.66×10^{-4}, 2.64×10^{-4}, 0.0047, and 0.031, respectively. The distributions of $\ln p$ values by the bootstrapping method are shown in Figure 5.8. Therefore, the Monte Carlo sampling and bootstrapping methods have the same conclusions.

TABLE 5.7: Summary results of the median p values obtained by Monte Carlo sampling (MC Sampling), bootstrapping, and the Markov chain Monte Carlo with kernel density estimation (MCMC+KDE).

	median p value		
	MC Sampling	Bootstrapping	MCMC+KDE
SVM vs GP	0.12	0.12	0.15
SVM vs FLDA_1	4.08×10^{-5}	3.33×10^{-5}	1.51×10^{-4}
SVM vs FLDA_2	7.84×10^{-4}	6.66×10^{-4}	0.002
GP vs FLDA_1	2.99×10^{-4}	2.64×10^{-4}	6.65×10^{-4}
GP vs FLDA_2	0.0054	0.0047	0.0092
FLDA_1 vs FLDA_2	0.033	0.031	0.049

By using MCMC with KDE, the corresponding median p values, as shown in Figure 5.9, are 0.15, 1.51×10^{-4}, 0.002, 6.65×10^{-4}, 0.0092 and 0.049, respectively. Therefore, all of the proposed methods have the same conclusions, that is, only the null hypothesis on the pair of SVM and GP cannot be rejected, whereas other pairs can be rejected. Table 5.7 summarises the median p values obtained by Monte Carlo sampling, Bootstrapping and MCMC with KDE methods. It should be noted that MCMC with KDE produces the median p value as closely as the significance level, 0.05, in the case of FLDA_1 vs FLDA_2. This may affect the confidence to reject the null hypothesis when the obtained p value is almost equal to the significance level. In response to this case, Table 5.4 has been referred, and it was found that there are only four datasets: `chemo`, `MNIST`, `SCOP-CSHPVZ`, and `sonar`, in which FLDA_1 performs noticeably worse than FLDA_2,



(a) SVM vs GP

(b) SVM vs FLDA_1

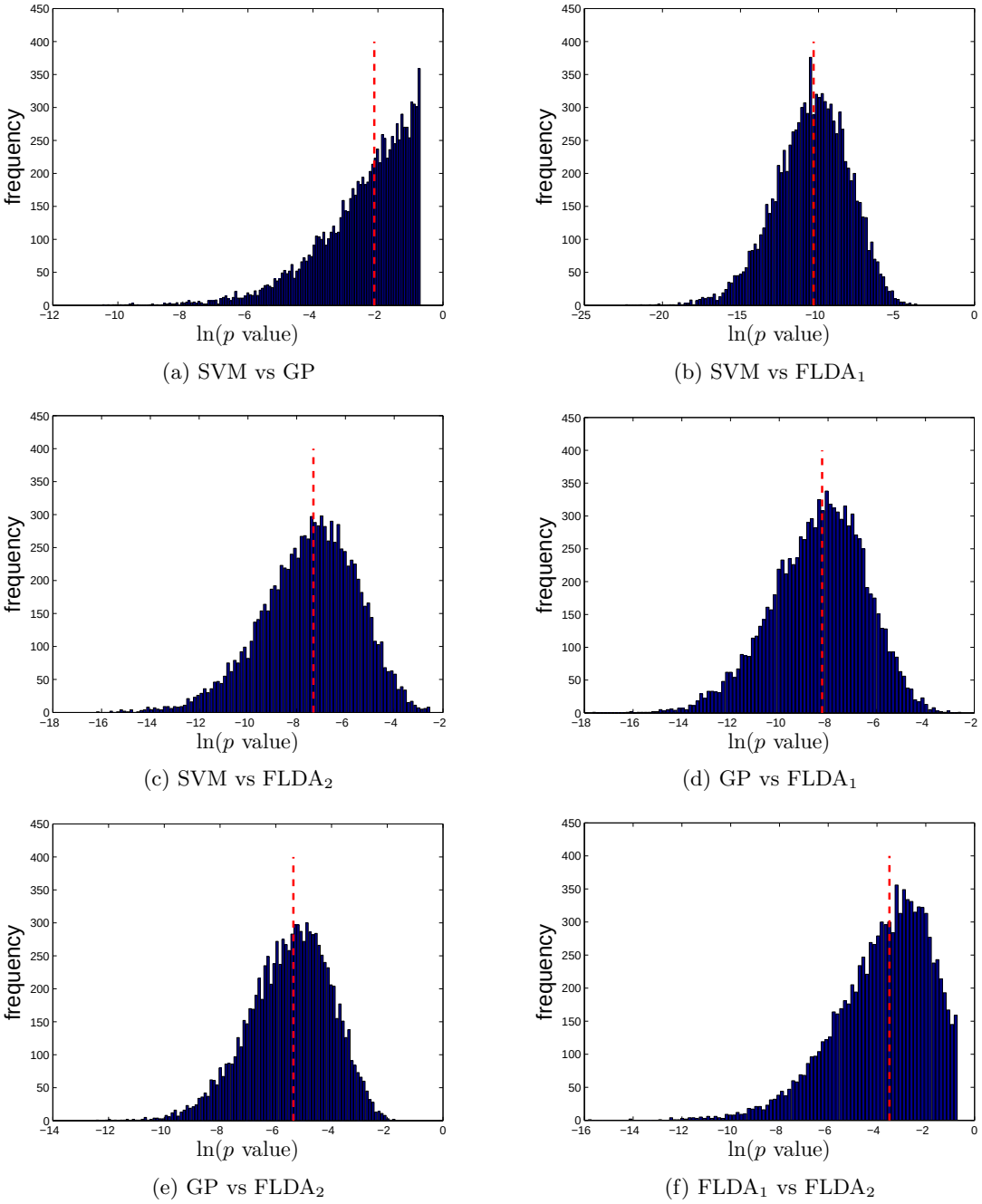
(c) SVM vs FLDA_2

(d) GP vs FLDA_1

(e) GP vs FLDA_2

(f) FLDA_1 vs FLDA_2

FIGURE 5.7: The distribution of $\ln p$ values using the Monte Carlo sampling method and running 10000 iterations. The dashed lines in the graphs represent the median of the $\ln p$ values. The corresponding median p values of the graphs (a) to (f) are 0.12, 4.08×10^{-5}, 7.84×10^{-4}, 2.99×10^{-4}, 0.0054, and 0.033, respectively.



(a) SVM vs GP

(b) SVM vs FLDA_1

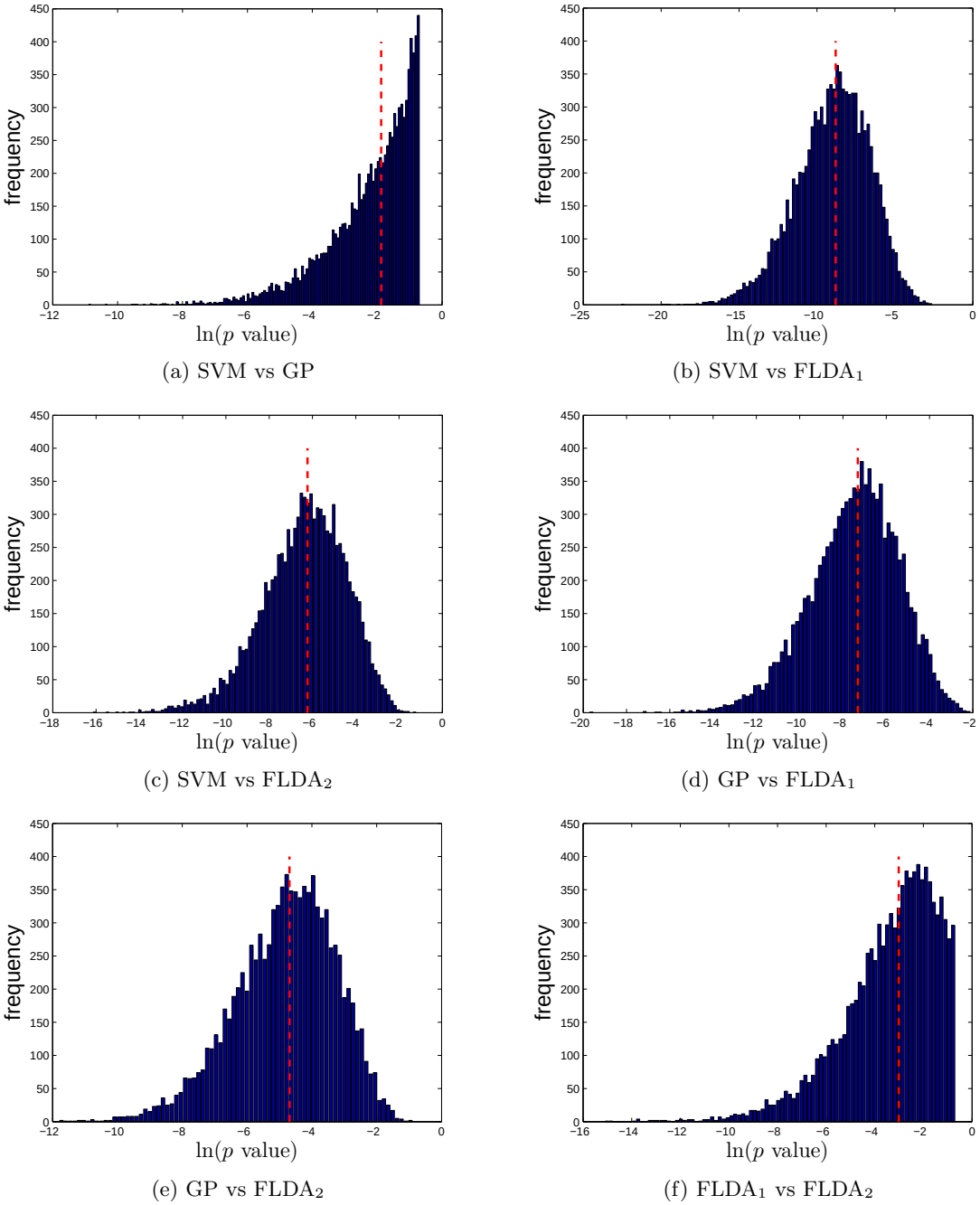
(c) SVM vs FLDA_2

(d) GP vs FLDA_1

(e) GP vs FLDA_2

(f) FLDA_1 vs FLDA_2

FIGURE 5.8: The distributions of $\ln p$ values by the bootstrapping sampling method, running 10000 iterations. The dashed lines in the graphs represent the median of the $\ln p$ values. The corresponding median p values of the graphs (a)to (f) are 0.12, 3.33×10^{-5}, 6.66×10^{-4}, 2.64×10^{-4}, 0.0047, and 0.031, respectively.



(a) SVM vs GP

(b) SVM vs FLDA_1

(c) SVM vs FLDA_2

(d) GP vs FLDA_1

(e) GP vs FLDA_2

(f) FLDA_1 vs FLDA_2

FIGURE 5.9: The distribution of $\ln p$ values by the Markov chain Monte Carlo with the kernel density estimation method running 10000 iterations. The dashed lines in the graphs represent the median of the $\ln p$ values. The corresponding median p values of the graphs (a) to (f) are 0.15, 1.51×10^{-4}, 0.002, 6.65×10^{-4}, 0.0092, and 0.049, respectively.

whereas the performances of both classifiers overlap for other datasets, due to the range of their uncertainties.

Comparing the results of the Wilcoxon matched-pairs signed-ranks and Friedman test to that of the proposed hypothesis testing, all of the tests mostly reject the same pairs. However, the main difference is the pair of SVM and GP in which the non-parametric tests reject the null hypothesis, but the proposed methods do not reject it. Considering only SVM and GP on the selected 27 datasets, the non-parametric tests accept that the performance of SVM is statistically better than that of GP, whereas the proposed methods accept that the performance of SVM is statistically comparable to the performance of GP. It is still arguable between SVM and GP users, but under the optimal conditions of each classifiers, it is believed that both are comparable. In this sense, the proposed sampling-based hypothesis testing which considers both score and uncertainty is a more appropriate test in classification than the standard test that considers only the score. It is not very surprising that the proposed hypothesis testing results agree with the traditional non-parametric statistical test results on many pairs because there are many noticeable average score difference among classifiers. Hence, the tests reject the null hypothesis due to the noticeable difference of the classifiers' performance. However, it is dubious whether the average accuracies and standard deviations among those classifiers are not very different; in other words, there are many common or overlapping scores when adding or subtracting the standard deviations from the average accuracies. For example, $91.80 \pm 2.94 \equiv [88.86, 94.74]$ vs $92.30 \pm 2.11 \equiv [90.19, 94.41]$, using both accuracies and standard deviations would be more appropriate than using only accuracies. As illustrated in Figure 3.1 and numerical results from Table 5.4, it is seen that the normal hypothesis tests can give a misleading conclusion when they consider only the average accuracies. Even though the accuracies are different, the uncertainty of those accuracies can cause the effect on the tests since the variability of the accuracies are much overlapping. It should be evident that the statistical test may reject the null hypothesis if the average scores from those classifiers are noticeably different, and the test should be reluctant to reject the null hypothesis when the scores are similar. However, a non-parametric test such as Wilcoxon matched-pairs signed-ranks test ignores this small difference in values and considers only the rank values. Thus, it often ends up rejecting the null hypothesis. The results on the synthetic datasets in section 5.1 and on real datasets in the experiments evidence the effectiveness of the proposed hypothesis testing in which the uncertainty from the cross-validation are taken into account.

5.3 Summary

We have demonstrated three sampling-based methods of hypothesis testing in this chapter. To illustrate the performance of the proposed hypothesis testing, twenty-seven

benchmark datasets were chosen and classified by SVM, GP, FLDAs under the same experimental settings, and then a formal statistical comparison was made. The Wilcoxon matched-pairs signed-ranks test, Friedman test as well as the proposed Monte Carlo sampling, MCMC with KDE and bootstrapping methods, which make use of additional information, that is, the standard deviation from 10CV, have been empirically performed. We have found that considering the variation in the average score is more sensible and benefits from the uncertainty to make a decision on the hypothesis testing. The results on both the synthetic and benchmark datasets show that the proposed hypothesis testing performs similarly and may be more appropriate than the general hypothesis testing when the average scores by classifiers overlap due to their uncertainties. Therefore, testing significant difference, which considers uncertainty, is a suitable option for users who want to make a formal statistical comparison of classification.

Chapter 6

Enhancements to SVM Classifier

In SVM, a small subset of training data known as support vectors is only required to construct the classifier; hence one can regard this as a sparse method equivalent to the sparse GP, which has the active set working the similar function of the support vectors. To find the support vectors, it is necessary to solve the quadratic programming in which all of the training instances are used and its complexity is in the degree of $\mathcal{O}(N^3)$, where N is the number of training instances, while the sparse GP runs in $\mathcal{O}(NM^2)$, where M is the number of instances in the active set. Moreover, the optimal parameters for SVM, such as C and γ when the RBF kernel is used, cannot be directly determined. In practice, they are normally tuned via a grid search strategy, which requires a great amount of time. As a result, SVM suffers from a longer running time during the training process especially in the case of problems with a large number of instances. Many researchers have been trying to speed up the training process by using many techniques and most of whom utilise the more complex mathematical form. A simple idea involving the data filtration or data selection is preferred over the more complex one in this chapter. Memory and running time are greatly reduced when many irrelevant data are removed. Besides, the data selection can be used as a pre-process and the complex method can be run later to get double action, which could prove beneficial. In this chapter, we explore three enhancements to the SVM classifier: (1) data subset selection to reduce the complexity of learning; (2) tree structure organisation of different classes in a multi-class problem; and (3) the performance of a specific kernel suitable for binary data. In Section 6.1 we give an overview of the existing data selection for SVM. Section 6.2 presents our novel methods of the data selection. Section 6.3 shows the performances of the data selection methods compared to the full SVM (SVM without data selection). The tree structure for a large multi-class is then presented in Section 6.4. Finally, in Section 6.5 we include a critical evaluation of the Tanimoto kernel against other kernels on binary data.

6.1 Data Selection

Although SVM has offered a good performance in classification problems, it is difficult to apply efficiently for large datasets due to high demand of memory and time complexity. The scalable issue comes from solving the QP objective function, which has the complexity of $\mathcal{O}(N^3)$. Many researchers have developed fast algorithms to solve the QP problem, such as chunking algorithm (Vapnik, 1982) and sequential minimal optimisation (Platt, 1999), in which the QP optimisation problem is decomposed into several smaller subproblems. Consequently, the complexity is reduced to the degree between linear and cubic in the size of the training data.

A simple way to reduce computational cost and speed up SVM is feeding smaller amounts of qualitative data rather than feeding the entire data to the QP. The qualitative data are support vectors used to create a classifier model with the maximum margin hyperplane. The problem is how to determine which data should be selected as the potential representatives. Given that many factors such as the types of kernel and kernel parameters affect the number of support vectors, the problem is modified to find out which data that are more likely to be support vectors. In other words some data are discarded regarding some criteria. In literature, there are two mainstreams of data selection that are: (1) clustering-based approach and (2) k-nearest-neighbor-based approach.

6.1.1 Clustering-based approach

A variation of clustering algorithms can be used in this approach, but k-means clustering is mostly used. K-means clustering partitions the data into k groups and the centre of the group can be regarded as the representative of the group. Almeida et al. (2000) employed k-means clustering to arrange data into k groups, and then the data of each homogeneous group (data belonging to the same class) are removed except for the centre points of the groups. These centre points and all data in heterogeneous groups (many classes in the same group) are treated as practical training data. However, they did not suggest how to choose the value of k. In their procedure, the initial k centre points are randomly selected; hence, different initial values may lead to different results. In addition, the location or shape of the hyperplane may be misleading since the calculated centre points at the final step often may not be the instances of the original training data. Similar to Almeida et al., Songfeng et al. (2003) also applied k-means clustering. However, they split the training data into two groups of positive and negative class data and then ran the clustering on the positive and negative groups separately, finally feeding the centre point of each cluster into the reduced SVM, called RSVM (Lee and Mangasarian, 2001), in which a fixed percentage ($1\% - 10\%$) of the training data is randomly selected, and these selected data are treated as the support vectors.

Boley and Cao (2004) proposed an algorithm called ClusterSVM, which trains SVM using adaptive clustering. ClusterSVM initially runs k-means clustering on both positive and negative data separately, feeds the representative of each cluster to an initial SVM and then approximately partitions the training data into support vector and non-support vector clusters. A cluster is defined as a support vector cluster if the cluster contains data with a functional margin less than or equal to 1 for positive class data or larger than -1 for negative class data. It is defined as a non-support vector cluster if the cluster contains data with a functional margin larger than 1 for positive class data or less than or equal to -1 for negative class data. Any cluster consisting of data with a functional margin of both less than and larger than 1 is split into two subclusters (support vector subcluster and non-support vector subcluster with an analogous definitions to the aforementioned ones). Consequently, the representative point of each support vector cluster and all data in the non-support vector clusters are used as the final selected training data. The drawback of Boley and Cao's approach is that the initial SVM is required, that is, two SVM classifiers are employed. The quality of the selected training data also depends on the initial SVM with well-tuned kernel and trade-off parameters.

A similar idea of ClusterSVM known as crisp clusters was introduced by Koggalage and Halgamuge (2004). They employed k-means clustering to identify crisp clusters that contain data with the same class. A non-crisp cluster is broken and a new crisp cluster with the same centre but a smaller radius is formed. There is another version of identifying crisp clusters. It works in the opposite way by starting with the centres and readily increasing the radius. Some data are removed according to a threshold sample percentage, which is defined by the ratio of the number of samples within the inner crisp cluster to the number of samples within the outer crisp cluster. In their work, a crisp cluster is set as a smaller threshold for the small radius and a higher threshold for the larger radius Obviously, the performance mainly depends on the location of the cluster centres, the threshold and the assigned increasing or decreasing of the one-unit radius. These parameters vary and need to be found on a trial and error basis.

Cervantes et al. (2008) applied a concept of minimum enclosing ball clustering (MEB), which partitioned the training data and then identified support vectors with SMO while removing the balls containing non-support vectors. The l ball centres are randomly chosen with an initial radius of r. The balls increase the radius if any data exist outside of the balls until they cover all data. The centres of the balls containing the same class are selected; otherwise, all data inside the balls are used in the first stage SVM with the SMO algorithm to identify the support vectors. Then the data inside the balls whose centres are support vectors are retrieved and used in the second stage SVM. Therefore, their method requires two SVM classifiers in order to select a subset of training data, and performance is mainly based on the quality of the first SVM and the choices of l and r. The number of balls, l, could be estimated by cross-validation; however, the radius of the balls is the crucial parameter.

The complexity of the clustering-based approach depends on which clustering algorithm is applied. For instance, the complexity of k-means algorithm is $\mathcal{O}(tkdN)$ (Duda et al., 2001, Chapter 10), where t and d are the number of iterations and features, respectively, whereas the complexity of hierarchical clustering is at least $\mathcal{O}(N^2)$. Nonetheless, the clustering-based approach, e.g., ClusterSVM algorithm, has a number of parameters for tuning such as the number of clusters, number of iterations and the radius distance. Some algorithms require more than one stage of SVM. These parameters cause unstable results and sometimes fail to reproduce the same results.

6.1.2 K-nearest-neighbor-based approach

The k-nearest-neighbor-based approach is an instance-based learning algorithm in which a distance function such as Euclidean distance is used for measuring the distance between an instance and its neighbors. The k nearest neighbors are then retained, leading to both less memory and storage for later running the SVM. The training data that are retained should be located near the boundary sometimes known as border point data because they are likely to be support vectors. Shin and Cho (2002) proposed an algorithm that provides training data that are likely to be located near the boundary based on what they call Neighbors_Entropy and Neighbors_Match. They defined Neighbors_Entropy($\boldsymbol{x}$) as the entropy of training data $\boldsymbol{x}$'s k-nearest neighbors' class labels and assumed that data $\boldsymbol{x}$ would be located near the boundary when the Neighbors_Entropy($\boldsymbol{x}$) is a positive value. To calculate the Neighbors_Entropy, the label probabilities P_j over J classes are computed by $P_j = \frac{k_j}{k}, \forall j$, where k_j is the number of k-nearest neighbors of data $\boldsymbol{x}$ that belongs to class j. Thus, the Neighbors_Entropy is calculated by $\sum_{j=1}^{J} P_j \cdot log_J \frac{1}{P_j}$. Finally the Neighbors_Match is computed by the ratio of $\boldsymbol{x}$'s neighbors whose label is the same as $\boldsymbol{x}$'s, that is, $\frac{k_j}{k}$ if data $\boldsymbol{x}$ belongs to class j. Although this method may reduce the data size, the reduction rate depends on the value of k. After all, the Neighbors_Match needs to be compared with a threshold, which they set as 0.5. This method is sensitive to the threshold, especially in larger dimensional problems. Another limitation is that the classes must be overlapped; otherwise, there will be a problem in the denominator of computing the label probability since P_j must not be zero.

Instance-based (IB) algorithms such as IB1-IB3 (Aha et al., 1991) can significantly reduce the number of data, but the SVM performance on these reduced data significantly degrades, as many support vectors are also removed during the data selection by the IB algorithms. Kim and Park (2004) came up with the hybrid between a kernelised instance based (KIB2) algorithm and what they called an ionic interaction (IoI) model. The KIB2 algorithm is IB2 in the feature space rather than input space. Given that KIB2 does not provide sufficient data to improve the accuracy, the IoI model is used to determine more candidate support vector data. The IoI model is imitated from the Coulomb potential energy between two ions, which is proportional to the product of

those two ions' charges and inversely to the Euclidean distance between them. The IoI model first calculates the potential energy for all data points, where the Euclidean distance between two different class data is computed in the feature space. The data in which the potential energy is less than a pre-defined threshold are chosen as candidate support vectors. Nonetheless, Kim and Park applied their algorithm only to binary problems and did not suggest how to set the threshold. Therefore, the performance of their method concerning the multi-class dataset is dubious because the threshold of each pair class may differ rather than one fixed threshold existing for every pair class.

A method called neighbor enemy distance or NED is the k-nearest-neighbor approach implemented by Ou et al. (2003) in order to expedite selecting the kernel parameters. This method greatly reduces the number of data when there are a lot of data of the same class located in a nearby region. The reduction rate depends on the k value. The method computes the distance of the nearest neighbor of each data $\boldsymbol{x}$, where the nearest neighbor's class is not the same as the class of data $\boldsymbol{x}$. Then the data are sorted in descending order by distance. The k-nearest neighbors of each data in the sorted set are determined, and if any data point whose class label is the same as those of all of the k-nearest neighbors, the data point is then removed. Ou et al. (2006) subsequently developed a method based on kernel density estimation (KDE) to eliminate redundant data. The method estimates a probability distribution for each class and then sorts the data in descending order by associated density function value. A data point is considered redundant if its density function value is either greater than a threshold or the k-nearest neighbor's class labels and the class label of the data are the same. Although the KDE method may reduce a large number of data, it is more complex and requires a lot of training time to obtain its optimal value. Moreover, the good feature selection needs to be employed in case of high dimensional problems.

Compared to the clustering-based approach, the k-nearest-neighbor-based approach runs slower as its complexity is $\mathcal{O}(N^2)$ due to the all pairwise computation of the distance function in the Neighbors_Entropy and NED algorithms except for IB algorithms in which the comlexity is $\mathcal{O}(N)$. However, it is more stable than the clustering-based approach. Another type of data selection is randomly selecting data, that is, simply choosing data based on uniform distribution. This random method normally produces different results when running at different times. Even though it is fast to run the random method, one disadvantage is that the suitable number of retained data is not known. Therefore, multiple iterations of random sampling data are run until the satisfactory result has been found. Instead of random selection, we propose a number of methods that automatically find out a number of retained data which are considered for SVM.

6.2 Proposed Data Selection Algorithms for SVM

Even though the clustering and k-nearest-neighbor-based approaches perform well, the researchers are uncomfortable with the unstable results and the tuning time that they have to adjust their parameters in the clustering methods and utilise a longer running time in the case of the k-nearest methods. We propose using certain algorithms that minimise the tuning time by not requiring any parameters and running between $\mathcal{O}(M^2)$ and $\mathcal{O}(N^2)$, where M is the number of selected data and $N > M$. The algorithms select a qualitative subset of training data before feeding to SVM. One of the straightforward ways to reduce parameter tuning is to follow Almeida et al. (2000)'s concept and merge with the process of determining the value of k in clustering instead of varying k repeatedly. Nonetheless, the running time of the merged method is not reduced. The remaining algorithms are run faster in either $\mathcal{O}(M^2)$ or $\mathcal{O}(D^3)$, depending on which value between M^2 and D^3 is larger, where D is the number of dimensions and is much less than N which is at least $N > D\sqrt{D}$. The algorithms focus on large problems in which the number of instances is much larger than the number of dimensions. Even though there are a large number of dimensions, an algorithm of dimension reduction techniques can be used to reduce D. For the next algorithm, we approximate each class as a Gaussian density with a different mean and covariance, and its parameters are estimated from the training data. Each class also has its own threshold to be compared to the density values of the data, and a subset of data is chosen if the density values are less than or equal to the corresponding thresholds. Then we introduce an algorithm called linear discriminant reduction (LDR) because its concept originates from Fisher's linear discriminant analysis. It retains the data in which the classes overlap on the projecting axis. The last algorithm is a hybrid of LDR and NED. It first runs LDR and is followed by NED to produce the final retained training data. We explain each algorithm in the following subsections.

6.2.1 Merged algorithm

In order to automatically determine the number of clusters, cluster centres and cluster members, we utilise affinity propagation clustering (Frey and Dueck, 2007) because the returned cluster centres are the instances in the training data, not the averaged data as the normal clustering methods return. However, it is not restricted to this clustering algorithm. Any algorithms that can return the number of clusters, cluster centres and their member can be used. Affinity propagation clustering or passing message algorithm is similar to the concept of message-passing in a graphical model in which each data is described as a node with edges or links to its neighbors. The connection edge between two nodes is used for communicating or exchanging messages. The nodes transmit and receive the messages recursively until converged clusters occur or maximum iterations are exceeded. The messages can be divided into two types: responsibility and availability.

The responsibility $r(i,k)$ shows how well the data k should be the cluster centre from the point of view of data i whereas the availability $a(i,k)$ shows how appropriate it is from the data k's viewpoint that the data k should be the cluster centre for data i. The affinity propagation algorithm is summarised and illustrated in Algorithm 6 and Figure 6.1, respectively.

Algorithm 6 Pseudocode of affinity propagation clustering (APC) from Frey and Dueck (2007)

Input: All training data $\boldsymbol{x}_i$
Output: l clusters with l input data voted as cluster centres and members in the clusters

1: initialise $r(i,j) \leftarrow 0, a(i,j) \leftarrow 0$
2: calculate similarity $s(i,k) \leftarrow -\|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2$, $s(j,j) =$ median of s
3: **repeat**
4: calculate responsibilities $r(i,k)$

$$r(i,k) \leftarrow s(i,k) - \max_{k' s.t.\ k' \neq k} \{a(i,k') + s(i,k')\}$$

5: calculate availabilities $a(i,k)$

$$a(i,k) \leftarrow \min \left\{ 0, r(k,k) + \sum_{i' s.t.\ i' \notin \{i,k\}} \max\{0, r(i',k)\} \right\}$$

6: update self-availabilities $a(k,k)$

$$a(k,k) \leftarrow \sum_{i' s.t.\ i' \neq k} \max\{0, r(i',k)\}$$

7: update data point k as the centre for data point i

$$\max_k a(i,k) + r(i,k)$$

8: **until** clusters no longer change or maximum iteration reached

Once the affinity propagation finishes, the cluster centres and the number of clusters are returned and then the Almeida's concept based on the clusters from the affinity propagation is performed. The class labels of all data in each cluster are examined and if all labels belong to the same class, only the centre point is chosen as the representative of that cluster. Otherwise all data in the cluster are retained. The pseudocode of the merged algorithm is shown in Algorithm 7.

6.2.2 Thresholded Gaussian

In the thresholded Gaussian algorithm, we approximate each class as a Gaussian distribution, and the approximated distributions have different means and covariances estimated

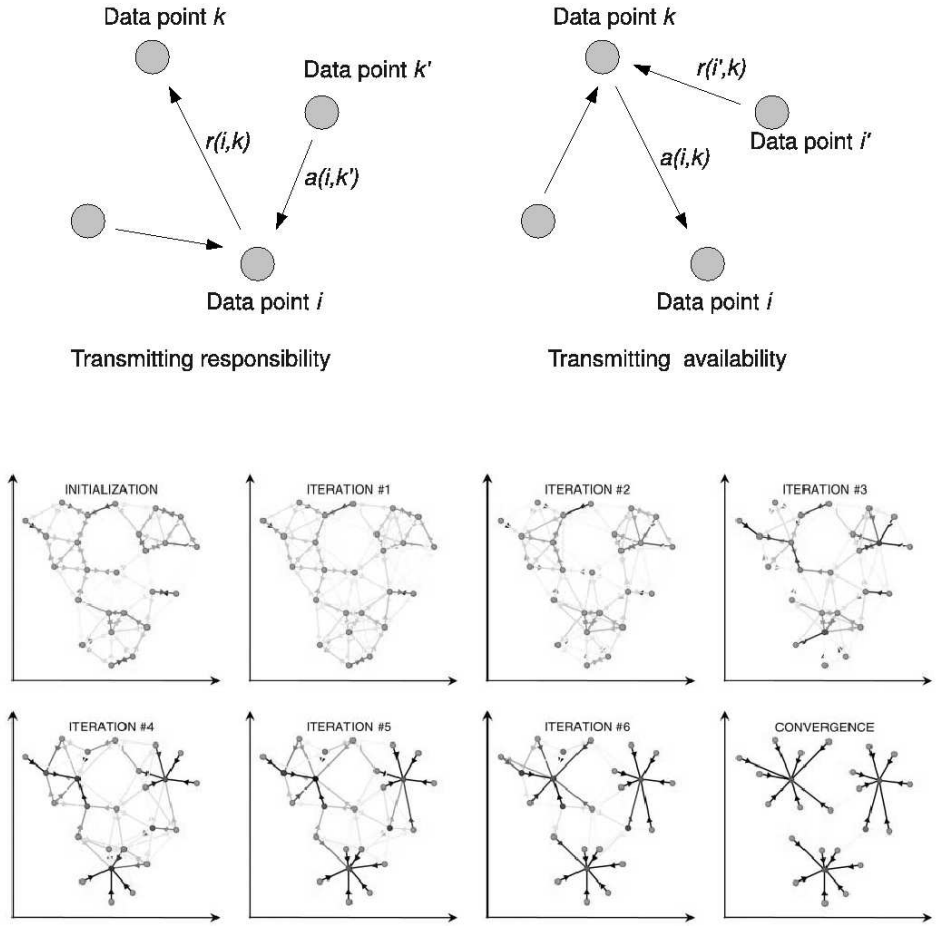


FIGURE 6.1: Top: Illustration of transmitting responsibility messages (left) and availability messages (right) where data k is the candidate of the cluster centre, Bottom: Illustration of affinity propagation – pictures from Frey and Dueck (2007)

Algorithm 7 Pseudocode of Merging APC with Almeida's approach (APCA)

Input: All training data $\boldsymbol{x}_i$
Output: Selected data $SDATA$

run affinity propagation clustering
for each cluster k **do**
  if all data have same class **then**
    remove all data except the cluster centre
    $SDATA_k \leftarrow$ centre point data
  end if
  $SDATA_k \leftarrow$ all data in the cluster
end for
$SDATA = \bigcup_k SDATA_k$

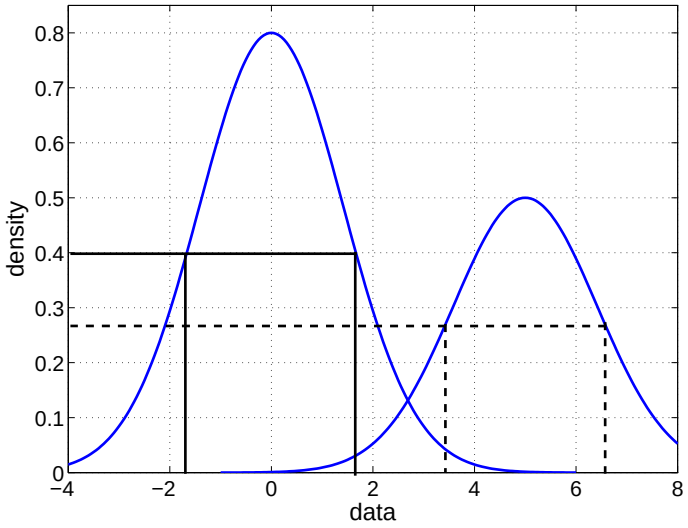


FIGURE 6.2: The Gaussian density approximation of two classes on 1D data illustrating the concept of the thresholded Gaussian algorithm where each class is a Gaussian distributed with its own threshold depicted by horizontal solid line for the first class and horizonal dashed line for the second class.

from the training data. The density values of the training data are then computed on a class-by-class basis. A threshold of each class is determined by averaging the density values from the corresponding class data. Data points with the density value lower than the corresponding threshold are selected as training data. We assume that these data are likely to be support vectors, as support vectors are likely to be located near the boundary and many of the overlapping data occur around this area. Figure 6.2 shows the density of simple 1D data of a two-class problem and demonstrates the concept of the thresholded Gaussian algorithm. The corresponding thresholds for the first and second class are indicated by the horizontal solid line and dashed line, respectively. In this example, data from the first class whose density values are greater than 0.4, or equivalently data with values in the range of -1.7 to 1.7 are discarded. Similarly, the data from the second class with values in the range of 3.5 to 6.6 are discarded. The threshold of each class is a parameter controlling the number of training data. If the threshold is very high, many data are chosen. On the contrary, many data are discarded if the threshold is very low. Empirically, the average density values of each class data is the recommended threshold for the corresponding class based on the reduction and accuracy rate.

Another possible version of the thresholded Gaussian algorithm is a mixture of Gaussian density. Rather than evaluating the data on the basis of class-by-class thresholds, the mixture of Gaussian density from all classes is calculated and one global threshold is established for selecting which data are used for the training data. We present both versions of the thresholded Gaussian integrated into Algorithm 8.

Algorithm 8 Pseudocode of class-by-class thresholded Gaussian/mixture of Gaussian density

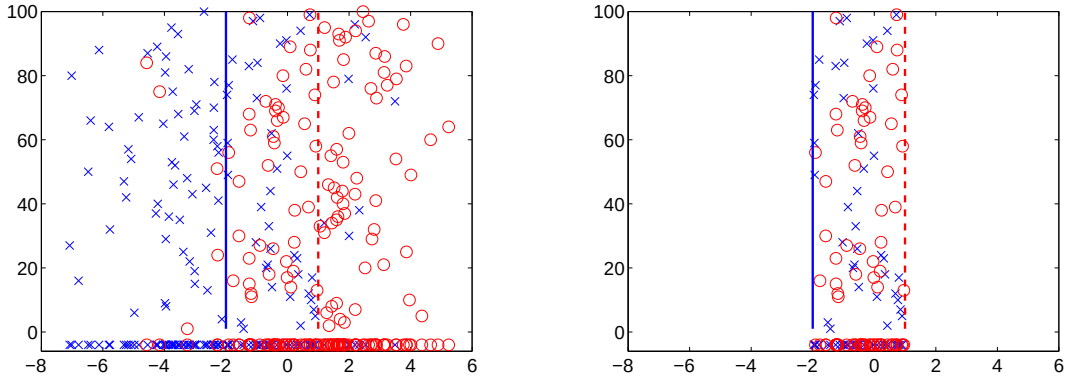
Input: All training data $\boldsymbol{x}_i$
Output: Selected data
Separate data by class
calculate mean ($\boldsymbol{\mu}_k$) and inverse of covariance ($\boldsymbol{\Sigma}_k^{-1}$) of each class k
if algorithm = class-by-class thresholded Gaussian **then**
  for all data $\boldsymbol{x}_i$ in class k **do**
    calculate density value $f_k(\boldsymbol{x}_i) = \exp\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\}$
    $threshold_k \leftarrow average(f_k(\boldsymbol{x}_i))$
    retain data $\boldsymbol{x}_i$ whose $f_k(\boldsymbol{x}_i) \leq threshold_k$
  end for
else if algorithm = mixture of Gaussian **then**
  calculate $f(\boldsymbol{x}_i) = \sum_k \exp\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\}$ for all data $\boldsymbol{x}_i$
  $threshold = average(f(\boldsymbol{x}_i))$
  retain data $\boldsymbol{x}_i$ whose $f(\boldsymbol{x}_i) \leq threshold$
end if

6.2.3 Linear discriminant reduction

We introduce the data reduction algorithm called linear discriminant reduction (LDR), as the concept of Fisher's linear discriminant analysis is applied to remove some redundant data from the original training data. Its concept is maximising the between-group mean and keeping the training data which fall between the group means. In other words, it attempts to project the data onto a separating line and preserve the overlapping projected data. We assume that the qualitatively candidate support vectors stand on the overlapping area of the projecting line. Figure 6.3 illustrates the concept of the LDR algorithm on a two-class synthetic data in which each class of 100 data points is generated from Gaussian with the same standard deviation, but the mean of the first class is -2 while the mean of the other class is 1. The solid line and dashed line in Figure 6.3 indicate the means of these two classes. LDR maintains the data within these lines, as shown in Figure 6.3(b), while the data outside are eliminated. The remaining data are used for the training step. In this example, there are 78 data left after running the LDR, compared to the original 200 data before running the LDR. Another variation of LDR is setting a threshold and keeping the data located in the range of the positive class mean and the threshold. The threshold is determined by the average of the positive and negative class mean. This scheme is referred to as LDR1Side or LDR1S in Algorithm 9 which shows the pseudocode of the LDR/LDR1S algorithm.

6.2.4 LDRNED

The last proposed algorithm is a combination of LDR and NED; hence, the name is LDRNED. We first apply the LDR algorithm to obtain the pre-selected data and then passes the pre-selected data through the NED algorithm, which finally produces the final



(a) Before removing data

(b) After removing data

FIGURE 6.3: Illustration of the LDR algorithm. Two synthetic classes are represented by crosses and circles where (a) is the data distribution before running the LDR and (b) is the data distribution after running the LDR. The solid line and dashed line are the means of those classes. The projections of the data on x-axis are shown at the bottom of the graphs.

Algorithm 9 Pseudocode of LDR/LDR1S

Input: All training data $\boldsymbol{x}_i$
Output: Selected data

for each class $i = 1, \ldots, k$ **do**
  split data into data of positive class ($\boldsymbol{D}_i$) and data of negative class ($\boldsymbol{D}_{j \neq i}$)
  calculate $\boldsymbol{\mu}_i = mean(\boldsymbol{D}_i)$ and $\boldsymbol{\mu}_j = mean(\boldsymbol{D}_j)$
  calculate $\boldsymbol{\Sigma}_i = cov(\boldsymbol{D}_i)$ and $\boldsymbol{\Sigma}_j = cov(\boldsymbol{D}_j)$
  calculate $\boldsymbol{w} = (\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j)^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T$
  calculate $\bar{w}_i = mean(\boldsymbol{w}^T \boldsymbol{D}_i^T)$ and $\bar{w}_j = mean(\boldsymbol{w}^T \boldsymbol{D}_j^T)$
  calculate threshold $= 0.5\boldsymbol{w}(\boldsymbol{\mu}_i + \boldsymbol{\mu}_j)^T$
  if algorithm = LDR **then**
    retain data in which the corresponding weights are between $\bar{w}_i$ and $\bar{w}_j$
  else if algorithm = LDR1S **then**
    retain data in which the corresponding weights are between $\bar{w}_i$ and threshold
  end if
end for

selected training data. The concept of NED has been described in Section 6.1.2. The logic behind this combination is that the data in which the class labels are overlapping are very likely to be support vectors, and the data which are far from the different classes and surrounded by data with the same labels are less likely to be support vectors. LDR does the work of retaining the projected data of the overlapping class, while NED eliminates the redundant data surrounded by the same class neighbors and retains the data that overlaps with the class. Similarly, the combination of LDR1S and NED is simply called LDR1SNED. We suggest using LDR1SNED for the problems in which the number of instances is too large to be fitted in memory due to its reduction rate efficiency. There is another modified-version of LDR1SNED which increases the accuracy rate compared to the normal LDR1SNED algorithm, as a result of retrieving more candidate data.

The modification of LDR1SNED is referred to as LDR1S2NED. However, given that it requires more training time than the usual one, it is suggested to use LDR1S2NED where the selected data are not sufficient to build a strong classifier. More discussion on LDR1S2NED is deferred to Section 6.3.2.

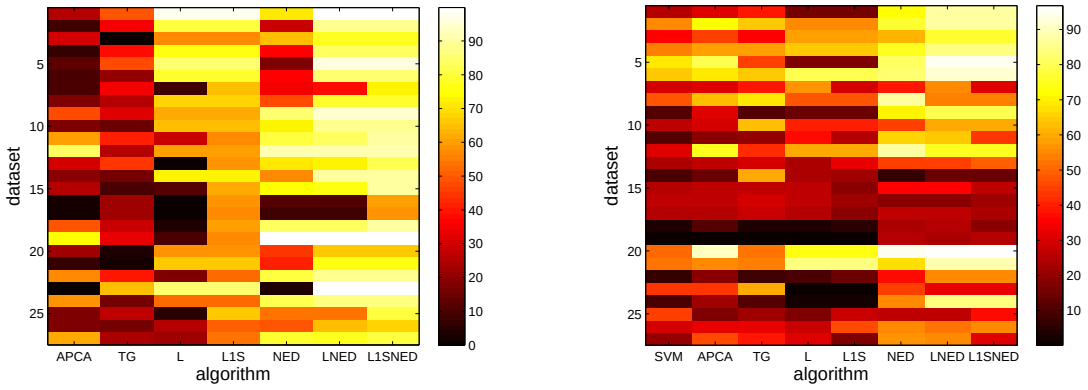
6.3 Experiments and Results

To demonstrate and compare the performances of the proposed data selection methods with SVM, the same benchmark datasets employed in Chapter 5 were used with the same experimental settings described in Section 5.2. Fifteen datasets are binary problems while twelve datasets are multi-class problems. The datasets vary from 2 to 27 classes and dimensionality varies from 2 to 22283. The characteristics of those datasets are reproduced in Table 6.1 for the convenient observation here.

TABLE 6.1: Summary of the datasets used in the experiments.

dataset	#class	#dim	#data		dataset	#class	#dim	#data
banana	2	2	5300		satimage	6	36	6435
breast cancer	2	9	277		SCOP-CSHPVZ	27	125	698
chemo	2	992	5747		SCOP-CSH	27	62	698
diabetes	2	8	768		segment	7	19	2310
flare-solar	2	9	1066		shuttle	7	9	58000
German	2	20	1000		sonar	2	60	208
glass	6	9	214		splice	2	60	3175
heart	2	13	270		thyroid	3	5	215
image	2	18	2310		titanic	2	3	2201
ionosphere	2	34	351		twonorm	2	20	7400
iris	3	4	150		vowel	11	10	990
lung cancer	2	22283	107		waveform	3	21	5000
MNIST	10	784	3500		wine	3	13	178
ringnorm	2	20	7400					

We compare the results of the proposed data selection algorithms with normal full SVM, which does not discard any data points. An overview of the results is summarised in Figure 6.4–6.6. Figure 6.4(a) presents the plot with a colour map of the reduction rate, while Figure 6.4(b) shows the colour map of the SV density, which is the ratio of the number of SVs to the number of selected data given by the proposed algorithms. Figure 6.5 and 6.6 display an overview of the comparisons among the algorithms measured on accuracy, standard deviation, the number of selected data and the number of support vectors, respectively. Table 6.2 to 6.4 show the numerical results. Table 6.2 shows the comparison results of affinity propagation clustering merged with Almeida's method and thresholded Gaussian method against the full SVM, while Table 6.3 compares LDR, LDR1S against the full SVM. The performance of NED, LDRNED, and LDR1SNED is shown in Table 6.4. The tables report the corresponding average accuracy, the number of selected data and the number of support vectors. These results are averaged on



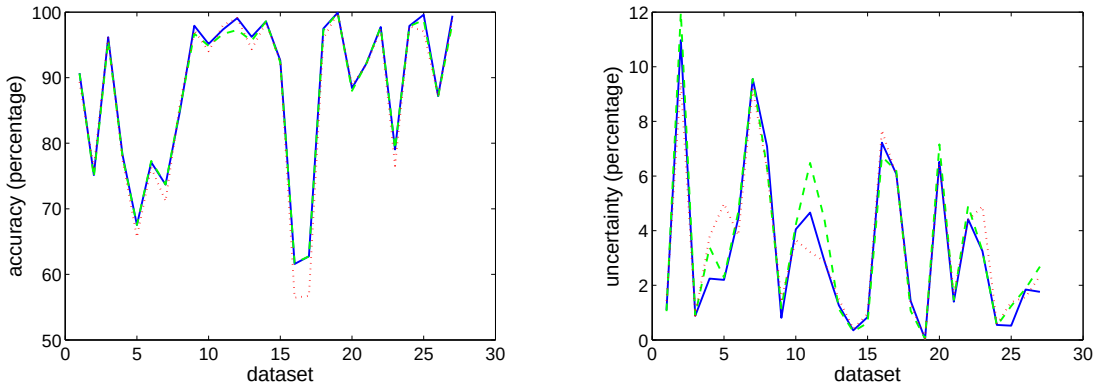
(a) Reduction rate (b) Ratio of $\frac{\#\text{SVs}}{\#\text{selected data}}$

FIGURE 6.4: Colour map of (a) reduction rate and (b) ratio of the number of support vectors to the number of selected data among all algorithms in which TG and L denote ThresGaus and LDR in the subfigures.

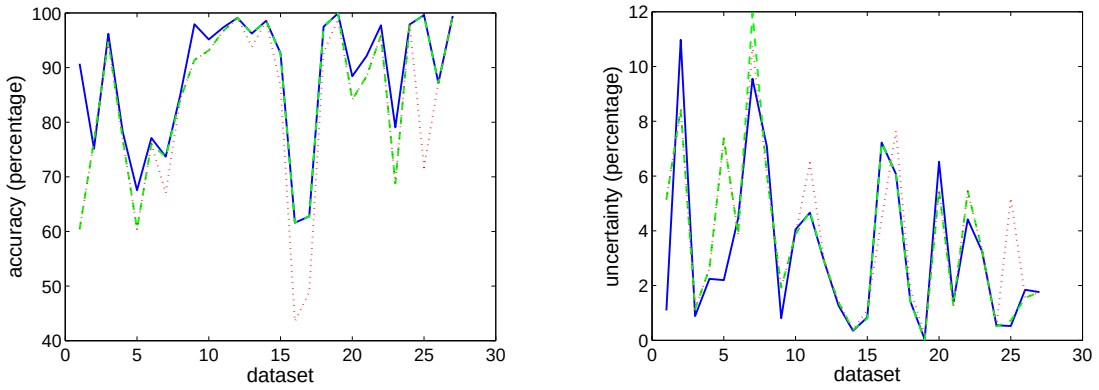
10CV. The optimal trade-off and kernel parameters are shown in Table 6.5. It should be noted that there is a problem, particularly in the `lung cancer` dataset, due to its large number of dimensions causing an out of memory error in MATLAB when calculating the covariance or invert matrix operation. Therefore, a feature selection based on the Fisher ratio is used to select a number of strong features, thus avoiding the memory problem due to the large dimension of data points when applying the proposed algorithms. The score of the dth feature is calculated by equation (6.1), in which μ, n_i, μ_i, σ_i^2 are the global mean, the number of instances, mean and variance in class i corresponding to the dth feature, respectively.

$$F_d = \frac{\sum_{i=1}^{c} n_i(\mu_i - \mu)^2}{\sum_{i=1}^{c} n_i \sigma_i^2} \tag{6.1}$$

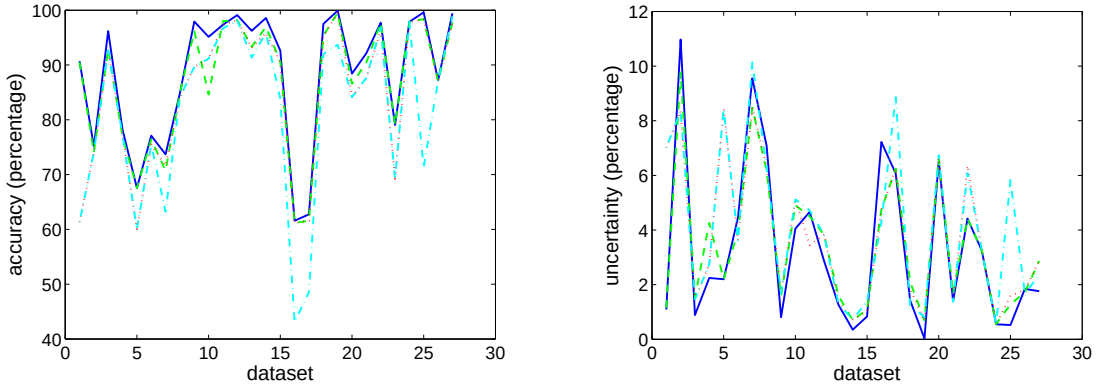
As there are a total of 107 data for the `lung cancer` dataset or approximately 11 data points per fold for 10CV, the first 13 strongest features selected by the aforementioned Fisher ratio were chosen in this dataset to circumvent the memory error and avoid a large dimension problem but small data points. Note that we use One-versus-All scheme for the full SVM classifier in the experiment. The RBF kernel is used for all SVM-based classifiers.



(a) full SVM vs APCA vs ThresGaus

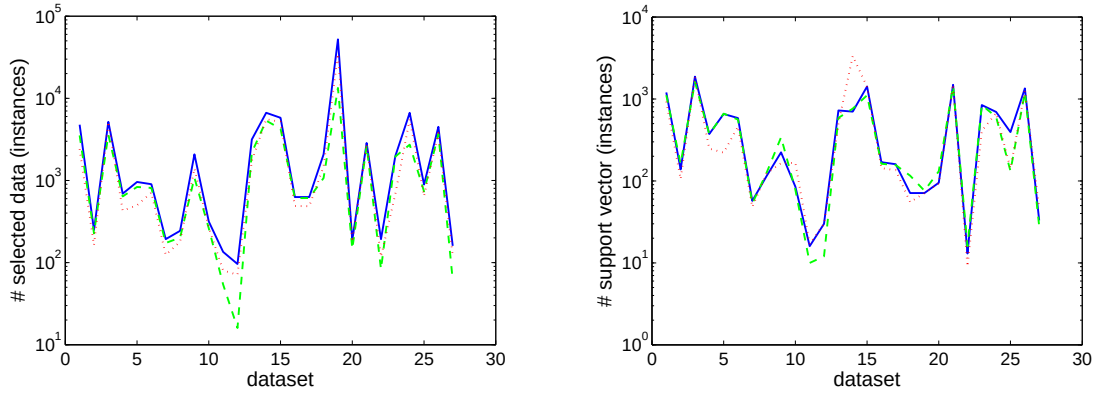


(b) full SVM vs LDR vs LDR1S

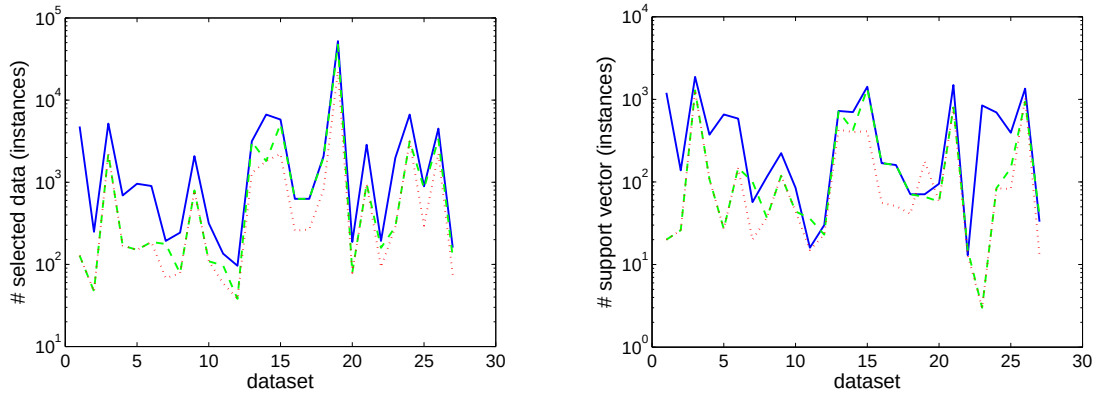


(c) full SVM vs NED vs LDRNED vs LDR1SNED

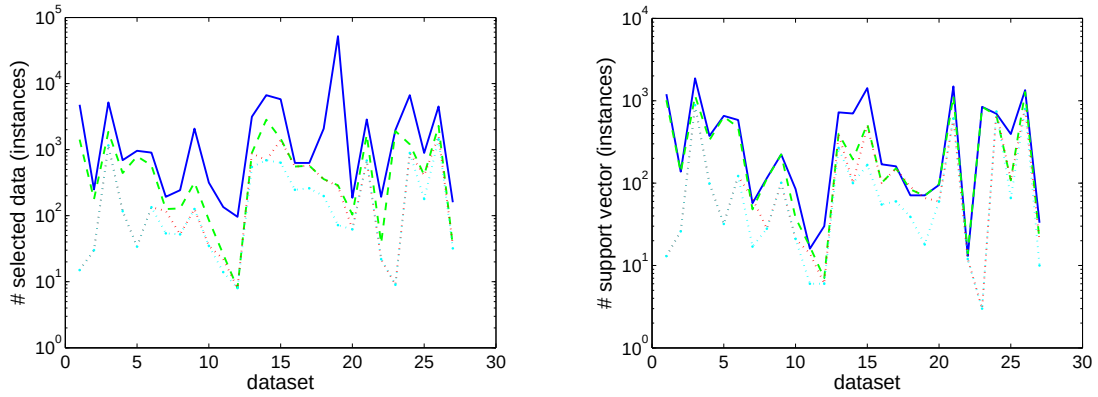
FIGURE 6.5: Accuracy and uncertainty comparison results on full SVM (blue solid line) against (a) APCA (green dashed line) and ThresGaus (red dotted line); (b) LDR (green dashed line) and LDR1S (red dotted line); (c) NED (green dashed line), LDRNED (red dotted line) and LDR1SNED (cyan dash-dot line).



(a) full SVM vs APCA vs ThresGaus



(b) full SVM vs LDR vs LDR1S



(c) full SVM vs NED vs LDRNED vs LDR1SNED

FIGURE 6.6: Comparison of the number of selected data and support vectors on the full SVM (blue solid line) against (a) APCA (green dashed line) and ThresGaus (red dotted line); (b) LDR,(L), (green dashed line) and LDR1S (red dotted line); (c) NED (green dashed line), LDRNED (red dotted line) and LDR1SNED (cyan dash-dot line).

Table 6.2: Comparison results among SVM, affinity propagation clustering merged with Almeida's method, and thresholded Gaussian.

dataset	full SVM			APCA			ThresGaus		
	acc	#train	#SVs	acc	#train	#SVs	acc	#train	#SVs
banana	90.70 ± 1.09	4770	1200	90.64 ± 1.05	3534	1113	89.32 ± 1.44	2403	927
breast cancer	75.14 ± 10.97	249	138	74.83 ± 11.98	227	162	76.21 ± 9.43	163	105
chemo	96.17 ± 0.89	5175	1872	95.68 ± 0.91	3600	1614	96.17 ± 0.89	5167	1872
diabetes	78.26 ± 2.24	691	374	78.00 ± 3.40	638	371	77.48 ± 3.78	429	248
flare-solar	67.54 ± 2.20	959	656	67.54 ± 2.28	833	661	65.66 ± 4.99	499	221
German	77.10 ± 4.46	900	584	77.30 ± 4.69	811	561	76.30 ± 3.83	712	460
glass	73.70 ± 9.55	193	57	73.70 ± 9.55	173	53	71.16 ± 9.17	124	48
heart	84.82 ± 7.08	243	115	85.19 ± 6.30	201	126	85.93 ± 6.25	179	124
image	97.92 ± 0.81	2079	223	96.71 ± 1.16	1078	341	97.23 ± 1.49	1403	164
ionosphere	95.14 ± 4.05	316	85	94.86 ± 4.22	261	78	94.02 ± 3.66	267	166
iris	97.33 ± 4.66	135	16	96.67 ± 6.48	54	10	98.00 ± 3.22	80	17
lung cancer	99.09 ± 2.87	96	30	97.27 ± 4.39	16	12	99.09 ± 2.87	71	30
MNIST	96.23 ± 1.27	3150	723	95.63 ± 1.14	2195	588	94.34 ± 1.50	1727	501
ringnorm	98.58 ± 0.35	6660	700	98.59 ± 0.31	5342	761	98.38 ± 0.41	5557	3348
satimage	92.62 ± 0.84	5787	1421	92.31 ± 0.62	4264	1108	92.53 ± 0.96	5176	1406
SCOP-CSH	61.59 ± 7.22	628	169	61.74 ± 6.71	609	162	56.45 ± 7.68	488	143
SCOP-CSHPVZ	62.75 ± 6.07	628	160	62.75 ± 6.19	610	156	56.73 ± 6.02	485	135
segment	97.49 ± 1.43	2079	71	97.23 ± 1.06	1069	116	95.97 ± 1.31	1475	55
shuttle	99.92 ± 0.03	52200	71	99.90 ± 0.04	13359	76	99.87 ± 0.05	34744	71
sonar	88.42 ± 6.52	187	95	88.00 ± 7.16	146	131	88.90 ± 6.47	179	93
splice	92.13 ± 1.40	2857	1490	92.16 ± 1.47	2639	1442	92.00 ± 1.68	2796	1502
thyroid	97.73 ± 4.42	193	13	97.73 ± 4.91	83	16	97.27 ± 4.39	117	9
titanic	79.06 ± 3.25	1981	843	79.06 ± 3.25	1953	842	76.56 ± 4.89	684	408
twonorm	97.89 ± 0.55	6660	695	97.84 ± 0.52	2726	603	97.85 ± 0.59	5555	672
vowel	99.60 ± 0.52	891	394	98.79 ± 1.24	734	132	97.07 ± 1.38	644	142
waveform	87.18 ± 1.85	4500	1346	87.08 ± 1.88	3662	1207	87.22 ± 1.56	3780	1258
wine	99.44 ± 1.76	160	33	98.33 ± 2.68	61	28	98.89 ± 2.34	121	46

Table 6.3: Comparison results among SVM, LDR, and LDR1S.

dataset	full SVM			LDR			LDR1S		
	acc	#train	#SVs	acc	#train	#SVs	acc	#train	#SVs
banana	90.70 ± 1.09	4770	1200	60.42 ± 5.15	128	20	60.42 ± 5.15	128	20
breast cancer	75.14 ± 10.97	249	138	76.61 ± 8.44	47	26	76.61 ± 8.44	47	26
chemo	96.17 ± 0.89	5175	1872	94.71 ± 1.14	2233	1286	94.71 ± 1.14	2233	1286
diabetes	78.26 ± 2.24	691	374	76.96 ± 2.63	169	108	76.96 ± 2.63	169	108
flare-solar	67.54 ± 2.20	959	656	60.44 ± 7.38	150	27	60.44 ± 7.38	150	27
German	77.10 ± 4.46	900	584	76.00 ± 3.86	187	148	76.00 ± 3.86	187	148
glass	73.70 ± 9.55	193	57	73.62 ± 11.95	177	101	67.03 ± 10.76	68	20
heart	84.82 ± 7.08	243	115	84.07 ± 6.06	78	37	84.07 ± 6.06	78	37
image	97.92 ± 0.81	2079	223	91.39 ± 1.92	795	119	91.39 ± 1.92	795	119
ionosphere	95.14 ± 4.05	316	85	93.15 ± 3.87	109	44	93.15 ± 3.87	109	44
iris	97.33 ± 4.66	135	16	96.67 ± 4.71	97	36	96.67 ± 6.48	59	15
lung cancer	99.09 ± 2.87	96	30	99.09 ± 2.87	38	23	99.09 ± 2.87	38	23
MNIST	96.23 ± 1.27	3150	723	96.20 ± 1.39	3102	713	93.83 ± 1.39	1298	430
ringnorm	98.58 ± 0.35	6660	700	98.54 ± 0.39	1808	425	98.54 ± 0.39	1808	401
satimage	92.62 ± 0.84	5787	1421	92.51 ± 0.79	5117	1352	86.61 ± 1.09	2214	411
SCOP-CSH	61.59 ± 7.22	628	169	61.59 ± 7.22	628	169	43.55 ± 4.43	257	56
SCOP-CSHPVZ	62.75 ± 6.07	628	160	62.75 ± 6.07	628	160	48.85 ± 7.64	266	51
segment	97.49 ± 1.43	2079	71	97.49 ± 1.41	2003	71	92.68 ± 1.86	818	41
shuttle	99.92 ± 0.03	52200	71	99.91 ± 0.02	47141	66	98.88 ± 0.20	22440	178
sonar	88.42 ± 6.52	187	95	84.14 ± 5.50	79	58	84.14 ± 5.50	79	58
splice	92.13 ± 1.40	2857	1490	88.41 ± 1.26	958	800	88.41 ± 1.26	958	800
thyroid	97.73 ± 4.42	193	13	95.91 ± 5.44	159	15	95.91 ± 5.44	92	13
titanic	79.06 ± 3.25	1981	843	68.79 ± 3.29	287	3	68.79 ± 3.29	287	3
twonorm	97.89 ± 0.55	6660	695	97.93 ± 0.51	3149	84	97.93 ± 0.51	3149	84
vowel	99.60 ± 0.52	891	394	99.49 ± 0.71	842	147	71.72 ± 5.19	296	82
waveform	87.18 ± 1.85	4500	1346	87.12 ± 1.56	3335	935	87.26 ± 1.57	2232	1041
wine	99.44 ± 1.76	160	33	99.44 ± 1.76	123	39	99.44 ± 1.76	75	13

Table 6.4: Comparison results among SVM, NED, LDRNED and LDR1SNED.

dataset	full SVM			NED			LDRNED			LDR1SNED		
	acc	#train	#SVs	acc	#train	#SVs	acc	#train	#SVs	acc	#train	#SVs
banana	90.70 ± 1.09	4770	1200	90.49 ± 1.16	1416	1015	61.34 ± 7.01	15	13	61.34 ± 7.01	15	13
breast cancer	75.14 ± 10.97	249	138	74.07 ± 9.77	178	135	74.07 ± 8.36	30	26	74.07 ± 8.36	30	26
chemo	96.17 ± 0.89	5175	1872	92.97 ± 1.53	1859	1150	92.69 ± 1.44	1154	883	92.69 ± 1.46	1154	883
diabetes	78.26 ± 2.24	691	374	76.96 ± 4.26	442	333	77.21 ± 2.75	117	99	77.21 ± 2.75	117	99
flare-solar	67.54 ± 2.20	959	656	67.54 ± 2.20	792	640	60.07 ± 8.43	34	32	60.07 ± 8.43	34	32
German	77.10 ± 4.46	900	584	76.40 ± 3.95	576	476	75.50 ± 3.63	134	122	75.50 ± 3.63	134	122
glass	73.70 ± 9.55	193	57	70.76 ± 8.48	126	48	71.64 ± 8.46	118	65	63.22 ± 10.14	54	17
heart	84.82 ± 7.08	243	115	84.81 ± 6.16	129	113	84.07 ± 6.31	52	28	84.07 ± 6.31	52	28
image	97.92 ± 0.81	2079	223	96.15 ± 1.57	318	222	89.65 ± 1.68	126	101	89.65 ± 1.68	126	101
ionosphere	95.14 ± 4.05	316	85	84.61 ± 4.91	86	38	91.14 ± 5.12	35	21	91.14 ± 5.12	35	21
iris	97.33 ± 4.66	135	16	98.00 ± 4.50	26	17	97.33 ± 3.44	22	14	96.67 ± 4.71	14	6
lung cancer	99.09 ± 2.87	96	30	98.18 ± 3.83	8	7	98.18 ± 3.83	8	6	98.18 ± 3.83	8	6
MNIST	96.23 ± 1.27	3150	723	93.31 ± 1.60	890	398	93.14 ± 1.39	881	394	91.34 ± 1.30	546	267
ringnorm	98.58 ± 0.35	6660	700	96.95 ± 0.69	2840	194	95.89 ± 0.73	689	100	95.89 ± 0.73	689	100
satimage	92.62 ± 0.84	5787	1421	90.64 ± 1.10	1458	522	90.43 ± 1.02	1394	491	83.76 ± 1.38	633	166
SCOP-CSH	61.59 ± 7.22	628	169	61.17 ± 4.75	553	101	61.17 ± 4.75	553	101	43.13 ± 4.30	247	55
SCOP-CSHPVZ	62.75 ± 6.07	628	160	61.59 ± 6.25	570	151	61.59 ± 6.25	570	151	48.56 ± 8.86	260	60
segment	97.49 ± 1.43	2079	71	95.37 ± 2.10	358	84	95.24 ± 1.84	356	90	91.99 ± 1.25	198	39
shuttle	99.92 ± 0.03	52200	71	99.52 ± 0.70	288	70	99.51 ± 0.67	280	66	93.70 ± 0.82	72	18
sonar	88.42 ± 6.52	187	95	86.57 ± 6.60	104	98	84.14 ± 6.73	62	60	84.14 ± 6.73	62	60
splice	92.13 ± 1.40	2857	1490	90.43 ± 1.67	1689	1147	87.65 ± 1.37	682	608	87.65 ± 1.37	682	608
thyroid	97.73 ± 4.42	193	13	97.73 ± 4.42	38	14	96.36 ± 6.36	22	12	96.82 ± 6.08	22	12
titanic	79.06 ± 3.25	1981	843	79.06 ± 3.25	1910	843	69.24 ± 3.26	9	3	69.24 ± 3.26	9	3
twonorm	97.89 ± 0.55	6660	695	97.91 ± 0.56	1204	662	97.89 ± 0.55	889	741	97.89 ± 0.55	889	741
vowel	99.60 ± 0.52	891	394	98.38 ± 1.28	412	108	98.28 ± 1.58	405	108	71.31 ± 5.84	179	66
waveform	87.18 ± 1.85	4500	1346	87.04 ± 1.72	2290	1275	87.20 ± 1.81	1605	835	87.12 ± 1.67	1427	794
wine	99.44 ± 1.76	160	33	97.78 ± 2.87	35	20	97.78 ± 2.87	36	20	98.89 ± 2.34	32	10

Table 6.5: Selected optimal trade-off and kernel parameter values of 2^C,2^γ on each method.

dataset	full SVM		APCA		ThresGaus		LDR		LDR1S		NED		LDRNED		LDR1SNED	
	C	γ	C	γ	C	γ	C	γ	C	γ	C	γ	C	γ	C	γ
banana	0	0	3	-1	-1	-1	6	-5	6	-5	5	-1	3	0	3	0
breast cancer	8	-8	-1	-3	4	-6	4	-8	4	-8	4	-6	0	-4	0	-4
chemo	4	-10	4	-10	4	-10	12	-10	12	-10	4	-10	12	-10	4	-10
diabetes	4	-5	4	-5	9	-8	1	-2	1	-2	8	-5	1	-2	1	-2
flare-solar	3	-4	1	-3	5	-4	12	1	12	1	5	-6	-1	0	-1	0
German	0	-4	0	-4	2	-5	1	-4	1	-4	0	-5	1	-4	1	-4
glass	5	1	5	1	6	1	2	3	12	-6	6	1	3	2	10	-1
heart	5	-10	0	-6	1	-7	4	-7	4	-7	2	-8	12	-4	12	-4
image	11	-3	7	-1	10	-5	4	-3	4	-3	5	-2	1	-3	1	-3
ionosphere	2	-4	2	-4	-1	-2	2	-4	2	-4	7	-6	7	-6	7	-6
iris	8	-5	12	-6	7	-5	1	-3	1	-1	1	-1	1	0	7	-3
lung	-1	-1	0	0	-1	-1	0	0	0	0	1	-1	12	-2	12	-2
MNIST	12	-7	1	-7	12	-7	12	-7	12	-7	12	-7	12	-7	12	-7
ringnorm	-2	-4	-1	-3	12	-2	-1	-3	-1	-3	8	-9	1	-4	1	-4
satimage	2	1	2	1	2	1	2	1	1	0	2	0	3	0	7	-7
SCOP-CSH	12	-2	12	-2	12	-2	12	-2	12	-3	4	-3	4	-3	12	-3
SCOP-CSHPVZ	3	-3	3	-3	3	-3	3	-3	3	-5	12	-3	12	-3	2	-4
segment	12	-1	4	0	12	-4	12	-1	9	-4	4	-1	3	-1	5	-2
shuttle	11	3	11	4	10	2	11	3	4	0	6	4	6	4	5	3
sonar	4	-2	12	1	4	-2	4	-2	4	-2	12	0	12	0	12	0
splice	1	-6	1	-6	2	-6	0	-6	0	-6	1	-6	1	-6	1	-6
thyroid	8	-2	5	-1	11	-2	2	0	3	-1	8	-3	2	0	2	0
titanic	10	-3	10	-3	-1	-4	12	-3	12	-3	10	-3	12	-6	12	-6
twonorm	-1	-6	-1	-5	-1	-6	12	-8	12	-8	-1	-4	-2	-8	-2	-8
vowel	12	2	3	1	12	1	12	1	3	1	12	1	12	1	3	1
waveform	1	-5	10	-10	6	-9	7	-8	4	-8	-2	-3	1	-4	4	-6
wine	1	-3	0	-2	0	-1	0	-2	12	-5	2	-3	2	-3	12	-6

Table 6.6: Summary of data reduction rate (%).

dataset	APCA	ThresGaus	LDR	LDR1S	NED	LDRNED	LDR1SNED
banana	25.91	49.62	97.32	97.32	70.31	99.69	99.69
breast cancer	8.84	34.54	81.12	81.12	28.51	87.95	87.95
chemo	30.43	0.15	56.85	56.85	64.08	77.70	77.70
diabetes	7.67	37.92	75.54	75.94	36.03	83.07	83.07
flare-solar	13.14	47.97	84.36	84.36	17.41	96.45	96.45
German	9.89	20.89	79.22	79.22	36.00	85.11	85.11
glass	10.36	35.75	8.29	64.77	34.72	38.86	72.02
heart	17.28	26.34	67.90	67.90	46.91	78.60	78.60
image	48.15	32.52	61.76	61.76	84.70	93.94	93.94
ionosphere	17.41	15.51	65.51	65.51	72.78	88.92	88.92
iris	60.00	40.74	28.15	56.30	80.74	83.70	89.63
lung	83.33	26.04	60.42	60.42	91.67	91.67	91.67
MNIST	30.32	45.17	1.52	58.79	71.75	72.03	82.67
ringnorm	19.79	16.56	72.85	72.85	57.36	89.65	89.65
satimage	26.32	10.56	11.58	61.74	74.81	75.91	89.06
SCOP-CSH	3.03	22.29	0.00	59.08	11.94	11.94	60.67
SCOP-CSHPVZ	2.87	22.77	0.00	57.64	9.24	9.24	58.60
segment	48.58	29.05	3.66	60.65	82.78	82.88	90.48
shuttle	74.41	33.44	9.69	57.01	99.45	99.46	99.86
sonar	21.93	4.28	57.75	57.75	44.39	66.84	66.84
splice	7.63	2.14	66.47	66.47	40.88	76.13	76.13
thyroid	56.99	39.38	17.62	52.33	80.31	88.60	66.60
titanic	1.41	65.47	85.51	85.51	3.58	99.55	99.55
twonorm	59.07	16.59	52.72	52.72	81.92	86.65	86.65
vowel	17.62	27.72	5.50	66.78	53.76	54.55	79.91
waveform	18.62	16.00	25.89	50.40	49.11	64.33	68.29
wine	61.88	24.38	23.13	53.13	78.13	77.50	80.00

Table 6.7: Summary of ratio of support vectors to training data size (%).

dataset	full SVM	APCA	ThresGaus	LDR	LDR1S	NED	LDRNED	LDR1SNED
banana	25.16	31.49	38.58	15.63	15.63	71.68	86.67	86.67
breast cancer	55.42	71.37	64.42	55.32	55.32	75.84	86.67	86.67
chemo	36.17	44.83	36.23	57.59	57.59	61.86	76.52	76.52
diabetes	54.12	58.15	57.81	63.91	63.91	75.34	84.62	84.62
flare-solar	68.40	79.35	44.29	18.00	18.00	80.81	94.12	94.12
German	64.89	69.17	64.61	79.14	79.14	82.64	91.04	91.04
glass	29.53	30.64	38.71	57.06	29.41	38.10	55.08	31.48
heart	47.33	62.69	69.27	47.44	47.44	87.60	53.85	53.85
image	10.73	31.63	11.69	14.97	14.97	69.81	80.16	80.16
ionosphere	26.90	29.89	62.17	40.37	40.37	44.19	60.00	60.00
iris	11.85	18.52	21.25	37.11	25.42	65.38	63.64	42.86
lung	31.25	75.00	42.25	60.53	60.53	87.50	75.00	75.00
MNIST	22.95	26.79	29.01	22.99	33.13	44.72	44.72	48.90
ringnorm	10.51	14.25	60.25	23.51	22.18	6.83	14.51	14.51
satimage	24.56	25.98	27.16	26.42	18.56	35.80	35.22	26.22
SCOP-CSH	26.91	26.60	29.30	26.91	21.79	18.26	18.26	22.27
SCOP-CSHPVZ	25.48	25.57	27.84	25.48	19.17	26.49	26.49	23.08
segment	3.42	10.85	3.73	3.54	5.01	23.46	25.28	19.70
shuttle	0.14	0.57	0.20	0.14	0.79	24.31	23.57	25.00
sonar	50.80	89.73	51.96	73.42	73.42	94.23	96.77	96.77

Continued on Next Page...

Table 6.7 – Continued

dataset	full SVM	APCA	ThresGaus	LDR	LDR1S	NED	LDRNED	LDR1SNED
splice	52.15	54.64	53.72	83.51	83.51	67.91	89.15	89.15
thyroid	6.74	19.28	7.69	9.43	14.13	36.84	54.55	54.55
titanic	42.55	43.11	59.65	1.05	1.05	44.14	33.33	33.33
twonorm	10.44	22.12	12.10	2.67	2.67	54.98	83.35	83.35
vowel	44.22	17.98	22.05	17.46	27.70	26.21	26.67	36.87
waveform	29.91	32.96	33.28	28.04	46.64	55.68	52.02	55.64
wine	20.63	45.90	38.02	31.71	17.33	57.14	55.56	31.25

6.3.1 Accuracy versus Reduction rate

The experimental results show that in most cases, SVM without removing any data points (full SVM) performs with higher accuracy than the proposed data selection approaches. The reason is that no support vectors are lost and the trade-off is longer training in parameter tuning with the entire training data. In certain datasets such as `breast cancer` and `heart`, some proposed selection approaches produce higher accuracy, even though many data points are excluded. These results support that the data size, trade-off and kernel parameters have an influence on the support vectors and the performance, since the number of data and optimal parameter values by the proposed approaches differ from the full SVM. It turns out that support vectors from the data selection approaches differ from the original full SVM, but they are expected to contain some common data points. This result reveals that the number of data is the upper bound for support vectors, and the shape of the hyperplane is restricted to support vectors as well as the optimal parameters. It is also observed that a larger number of support vectors do not always have higher accuracy than a smaller number of support vectors, even though the larger one is likely to give better accuracy.

Comparing the combination of affinity propagation clustering and Almeida's concept to the thresholded Gaussian algorithm, both methods perform similarly. When the accuracy level between both methods is very similar, the merged method has a higher reduction rate. The noticeable differences between them concern the three multi-class problems `MNIST, SCOP-CSH,` and `SCOP-CSHPVZ`. The average reduction rate on all datasets of the merged method and thresholded Gaussian method are 29 percent and 27.55 percent, respectively. Due to the lower reduction rate, there is a high probability that many support vectors are likely to be preserved, as shown in the Figure 6.6. Hence, the accuracy rate of both methods is similar to that of SVM. By using the hypothesis testing proposed in Chapter 5, it turns out that both methods are not significantly different, with the 0.05 significance level as shown in Table 6.8. Nonetheless, the merged algorithm requires a longer pre-process time, especially when there are a large number of instances, as it calculates the similarity matrix and repeats the operation of responsibility and availability many times. The running time complexity of the merged method is $\mathcal{O}(N^2)$ whereas the

thresholded Gaussian runs in $\mathcal{O}(D^3)$, where N is the number of training instances and D is the number of dimensions.

TABLE 6.8: Summary results of the proposed hypothesis testing on full SVM, APCA and thresholded Gaussian (TG) comparison.

	median p value		
	MC Sampling	MCMC+KDE	Bootstrapping
SVM vs APCA	0.23	0.46	0.23
SVM vs TG	0.17	0.38	0.16
APCA vs TG	0.22	0.44	0.21

As mentioned earlier, there are two versions of the thresholded Gaussian method: the class-by-class thresholded Gaussian and the mixture of thresholded Gaussian. Table 6.9 shows average accuracy and the corresponding number of reduced data size in both versions. Both versions produce similar results. However, in four cases, the `banana, glass, SCOP-CSH` and `SCOP-CSHPVZ` datasets, the accuracy rate of the class-by-class concept is higher than that of the mixture's concept. The number of reduced training data shows that the mixture one has a higher data reduction rate than the class-by-class concept, and in those four problems, the reduction rates of the mixture concept are quite high. This suggests that too high a reduction rate may produce lower accuracy, as many support vectors are lost. This suggestion is also clearly seen on the `banana` dataset to which the LDR family was applied. The proposed hypothesis testing cannot reject the null hypothesis as the median p values are 0.085, 0.11 and 0.082, respectively, indicating that both versions are not statistically different in terms of performance.

In most of the datasets, LDR and LDR1S perform similarly, but LDR1S reveals a higher reduction rate. This is caused by selecting only instances of class i rather than instances from both class i and j where class i is the positive class and j is the negative class in each round. Therefore, in terms of reduction rate, LDR1S is more suitable than LDR for data reduction on large datasets such as `shuttle` dataset. There are four datasets (`glass, SCOP-CSH, SCOP-CSHPVZ, vowel`) in which LDR1S performs quite differently from LDR. The reduction rates on those datasets are approximately 60 percent using LDR1S, and less than 9 percent using LDR. This implies that LDR did not lose many support vectors, as approximately 91 percent of those datasets were used for training. Compared to the full SVM, both LDR algorithms have lower performances on a subset of datasets; however, on another subset of datasets, their performances are comparable to the full SVM. We ran the proposed hypothesis testing, and SVM was not found to be significantly different from LDR, but different from LDR1S. The corresponding median p values are shown in Table 6.10. The difference is believed to result from the high reduction rate of LDR1S.

TABLE 6.9: Summary of the average accuracy when applying the class-by-class thresholded Gaussian (TG) and the mixture of thresholded Gaussian (MTG). Numbers in the parenthesis show the corresponding reduced training data size. The last column shows the original training data size.

dataset	TG	MTG	original size
banana	89.32 ± 1.44 (2403)	75.11 ± 5.63 (989)	4770
breast cancer	76.21 ± 9.43 (163)	75.85 ± 10.27 (124)	249
chemo	96.17 ± 0.89 (5167)	96.17 ± 0.89 (5170)	5175
diabetes	77.48 ± 3.78 (429)	77.74 ± 3.89 (303)	691
flare-solar	65.66 ± 4.99 (499)	67.54 ± 2.20 (331)	959
German	76.30 ± 3.83 (712)	76.40 ± 3.66 (642)	900
glass	71.16 ± 9.17 (124)	65.83 ± 14.66 (94)	193
heart	85.93 ± 6.25 (179)	85.93 ± 7.77 (144)	243
image	97.23 ± 1.49 (1403)	94.98 ± 2.17 (1088)	2079
ionosphere	94.02 ± 3.66 (267)	95.71 ± 3.63 (215)	316
iris	98.00 ± 3.22 (80)	96.00 ± 3.44 (49)	135
lung cancer	99.09 ± 2.87 (71)	99.09 ± 2.87 (67)	96
MNIST	94.34 ± 1.50 (1727)	96.23 ± 1.27 (3144)	3150
ringnorm	98.38 ± 0.41 (5557)	98.41 ± 0.48 (3803)	6660
satimage	92.53 ± 0.96 (5176)	92.54 ± 0.90 (4947)	5787
SCOP-CSH	56.44 ± 7.68 (488)	42.98 ± 9.27 (230)	628
SCOP-CSHPVZ	56.73 ± 6.02 (485)	35.10 ± 6.30 (193)	628
segment	95.97 ± 1.31 (1475)	94.81 ± 2.52 (1353)	2079
shuttle	99.87 ± 0.05 (34744)	99.86 ± 0.05 (23661)	52200
sonar	88.90 ± 6.47 (179)	88.90 ± 6.47 (180)	187
splice	92.00 ± 1.68 (2796)	91.97 ± 1.65 (2812)	2857
thyroid	97.27 ± 4.39 (117)	97.27 ± 3.18 (80)	193
titanic	76.56 ± 4.89 (684)	78.33 ± 2.52 (554)	1981
twonorm	97.85 ± 0.59 (5555)	97.85 ± 0.59 (5063)	6660
vowel	97.07 ± 1.38 (644)	91.41 ± 2.09 (508)	891
waveform	87.22 ± 1.56 (3780)	87.26 ± 1.69 (3419)	4500
wine	98.89 ± 2.34 (121)	98.33 ± 2.68 (98)	160

We ran NED algorithm in order to compare its results with the LDR family. The average reduction rates of LDR, LDR1S and NED are approximately 44 percent, 65 percent and 56 percent, respectively. In most of the datasets, NED and LDR1S perform comparably, except for `banana, SCOPs, titanic` and `vowel` in which NED performs much better than LDR1S. The reduction rates in Table 6.6 show that in those datasets, LDR1S has a much higher reduction rate than NED. In terms of complexity, NED runs in $\mathcal{O}(N^2)$, whereas the LDR family runs in $\mathcal{O}(D^3)$. Hence, the LDR family runs faster when it is applied to the problems of large instances and $N \gg D$. In order to take the strength points of both NED and the LDR family, the combination of them leads to LDRNED and LDR1SNED. Both of the combinations are comparable in performance and have a higher reduction rate than LDR and LDR1S. They are also faster than NED because the latter requires $\mathcal{O}(N^2)$, whereas LDRNED and LDR1SNED run in $\mathcal{O}(M^2)$,

where N and M are the number of instances and $M < N$, provided that the number of instances is much greater than the number of dimensions. Using the full SVM as the base classifier, LDR and NED do not statistically differ from the full SVM, whereas LDR1S, LDRNED and LDR1SNED statistically differ from the full SVM in terms of their average accuracies and uncertainties. Table 6.10 summarises the median p values of the proposed hypothesis testing on the NED and LDRs family against the full SVM which is our control or base classifier.

TABLE 6.10: Summary results of the proposed hypothesis testing on NED and LDRs family against the full SVM which is the base classifier.

	median p value		
SVM vs	MC Sampling	MCMC+KDE	Bootstrapping
LDR	0.057	0.064	0.052
LDR1S	0.0019	0.017	0.0018
NED	0.097	0.15	0.092
LDRNED	0.013	0.046	0.012
LDR1SNED	2.78×10^{-4}	6.60×10^{-3}	2.37×10^{-4}

On the other hand, LDR1SNED offers a high data reduction rate while maintaining an accuracy rate comparable to other methods on a subset of datasets. Furthermore, its ratio of support vectors to training data size is also high compared to that of SVM. This means that there are less useless data when solving the QP function and then speeding up the training phase. Table 6.6 and 6.7 summarise the data reduction rate and the ratio of support vectors to training data, respectively, among the algorithms in the experiments. These tables and Figure 6.4 show that the combination between the LDR family and NED results in a higher reduction rate and higher SV density among all algorithms. It is seen that the LDRs – LDR family and its combination with NED – perform poorly on the `banana` dataset. One reason is the insufficient amount of data selected by LDRs. As mentioned above, too high a reduction rate could result in the loss of many potential support vectors. To increase the sufficient selected data, a modification of LDR1SNED was applied to the datasets in which the accuracy rates of LDR1SNED are quite a bit lower than that of SVM. We present the modification of LDR1SNED in next subsection.

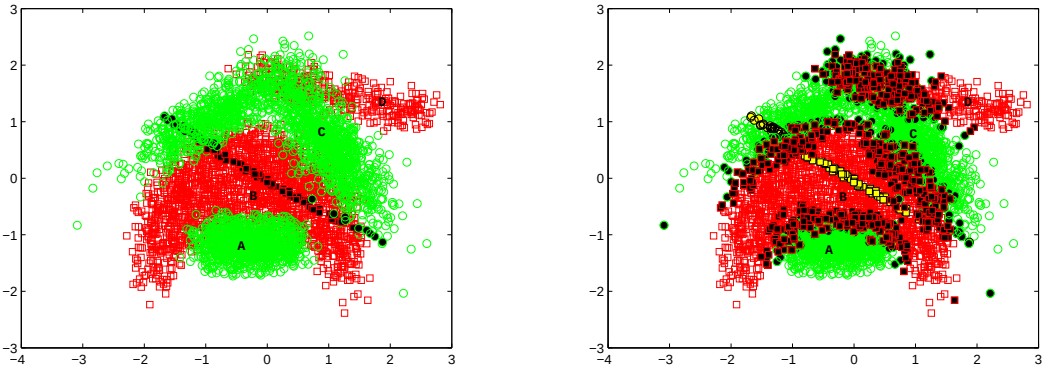
6.3.2 The Modification of LDR1SNED

As the main goal is to minimise the training size by discarding most redundant data due to the limited memory and expensive SVM computation, it is suggested to use LDR1SNED on large datasets. However, too many data are filtered out in some datasets. Hence, a modified version of LDR1SNED called LDR1S2NED should be used when there are not enough qualitative data selected by LDR1SNED. The pseudocode of

LDR1S2NED is shown in Algorithm 10. Conceptually, we add an extra subroutine to retain more data from the discarded data by passing the deselected data into another round of the NED. In the experiments, it is seen that LDRs perform worse than the others in terms of accuracy on the `banana` dataset. For this dataset, it turns out that a large number of input data are discarded by LDR1SNED, that is 99.69 percent, as shown in Table 6.6. In other words, there are only 15 data selected from the original 4770 training data for creating the classifier. Among the 15 data, thirteen are identified as support vectors, whereas there are 1200 support vectors among the 4770 training data from the full SVM. Compared to NED, the latter chooses 1416 data which contain 1015 support vectors. Therefore, LDR1SNED loses a lot of qualitative data in this particular dataset.

Algorithm 10 Pseudocode of LDR1S2NED

Input: All training data $\boldsymbol{x}_i$
Output: Selected data
1: run LDR1SNED
2: keep selected data in $SDATA1$ and deselected data in $DDATA$
3: run NED on $DDATA$ and keep selected data in $SDATA2$
4: selected data are $SDATA1 \cup SDATA2$



(a) (b)

FIGURE 6.7: Data distribution of the `banana` dataset. The squares, region B and D, denote the data of the first class and the circles, region A and C, represent the second class data. Left panel (a): the black shaded data are the data selected on the LDA axis. Right panel (b): the black and yellow shaded data are the data selected by LDR1S2NED.

Table 6.11 shows the average accuracy rate of LDR1S2NED on the subset of datasets in which LDR1SNED performs significantly differently from the full SVM. The performance of LDR1S2NED in terms of accuracy rate increases due to more qualitative data being retrieved. The number of selected data on the `banana` dataset increases to 1417 data points, of which 1015 are support vectors. Hence, more support vectors are obtained due to many potential data being retrieved. Why does LDR1SNED select a small portion of data points in this dataset? To answer this question, the data distribution of the `banana`

dataset in Figure 6.7(a) is plotted. It is found that the data clusters of each class are located next to each other. There is no data selected from region A and D projected on the LDA axis except for the black shaded areas in Figure 6.7(a). Since there is no selected training data in those regions, the classifier is likely to perform poorly when the test data come from region A and D. LDR1S2NED retrieves more data, as illustrated by the black and yellow shaded areas in Figure 6.7(b), and it is evident that some data from region A and D are added. The new model now has representatives in those regions, and the new classifier performs better than the previous one. To increase the accuracy rate, however, the trade-off is a longer running time, as LDR1S2NED runs in $\mathcal{O}(N^2)$. In practice, we suggest running LDR1SNED on a validation set, and if it performs poorly on the validation set then using LDR1S2NED. This strategy will have a complexity level between $\mathcal{O}(M^2)$ and $\mathcal{O}(N^2)$. The proposed hypothesis testing shows that LDR1S2NED is not significantly different from the full SVM as the median p value is greater than the 0.05 significance level shown in Table 6.12, whereas the full SVM and LDR1S2NED are different from LDR1SNED due to too many qualitative data loss in LDR1SNED.

Table 6.11: Comparison results on a subset of datasets among SVM, LDR1SNED and LDR1S2NED.

dataset	full SVM			LDR1SNED			LDR1S2NED		
	acc	#train	#SVs	acc	#train	#SVs	acc	#train	#SVs
banana	90.70 ± 1.09	4770	1200	61.34 ± 7.01	15	13	90.42 ± 1.09	1417	1015
flare-solar	67.54 ± 2.20	959	656	60.07 ± 8.43	34	32	65.96 ± 5.57	708	517
glass	73.70 ± 9.55	193	57	63.22 ± 10.14	54	17	72.74 ± 9.22	141	36
image	97.92 ± 0.81	2079	223	89.65 ± 1.68	126	101	96.41 ± 1.58	397	212
MNIST	96.23 ± 1.27	3150	723	91.34 ± 1.30	546	267	94.86 ± 1.04	1147	460
satimage	92.62 ± 0.84	5787	1421	83.76 ± 1.38	633	166	91.22 ± 1.23	1635	557
SCOP-CSH	61.59 ± 7.22	628	169	43.13 ± 4.30	247	55	61.31 ± 5.88	575	160
SCOP-CSHPVZ	62.75 ± 6.07	628	160	48.56 ± 8.86	260	60	62.47 ± 5.64	582	153
segment	97.49 ± 1.43	2079	71	91.99 ± 1.25	198	39	95.89 ± 1.45	454	82
sonar	88.42 ± 6.52	187	95	84.14 ± 6.73	62	60	88.90 ± 5.64	132	87
splice	92.13 ± 1.40	2857	1490	87.65 ± 1.37	682	608	91.44 ± 1.73	1855	1021
titanic	79.06 ± 3.25	1981	843	69.24 ± 3.26	9	3	79.06 ± 3.25	1872	709
vowel	99.60 ± 0.52	891	394	71.31 ± 5.84	179	66	99.09 ± 0.88	547	121

Table 6.12: Summary results of the proposed hypothesis testing on full SVM, LDR1SNED and LDR1S2NED comparison.

	median p value		
	MC Sampling	MCMC+KDE	Bootstrapping
SVM vs LDR1SNED	3.46×10^{-5}	2.62×10^{-4}	2.73×10^{-5}
SVM vs LDR1S2NED	0.20	0.43	0.20
LDR1SNED vs LDR1S2NED	1.31×10^{-4}	7.01×10^{-4}	9.40×10^{-5}

6.4 Tree Structure for Large Multi-Class Problems

In Ramanan et al. (2007), we proposed an unbalanced decision tree structure for classification problems known as UDT and we claimed that the testing time is shortened as the test data are classified during traversing the tree instead of waiting until the end of the tree. The details of the UDT concept are described in Chapter 2. The construction of the UDT, however, requires a great amount of time for training as a result of tuning the optimal parameters for a number of classifiers at every decision node level of the tree. For example, assume that a constant T is the running time for parameter tuning and there are 225 combinations of parameter C and γ tuning on SVM with the RBF kernel. In K-class problems, the UDT creates $K, (K-1), (K-2), \ldots, 3$ and 1 classifiers from the root node until the last node, respectively, and each classifier runs 225 times with the parameter tuning. Therefore, it uses $225T(\frac{K}{2}(1+K)-2)$ time units for tuning. This is too much tuning time when the number of classes is large.

In this section, we demonstrate a novel improvement to the UDT concept that addresses the above issue. The accelerated version of UDT is called "vine", and it requires less training time than the normal UDT, while its performance is still comparable to the normal SVM and UDT, as shown in Table 6.13. Table 6.13 shows the accuracy rate of vine compared to the SVM on the multi-class problems. It should be noted that vine, UDT and SVM produce the same accuracy rate in the two-class problems because vine and UDT have only one node and acts as an SVM. Table 6.14 shows that there is no significant difference among SVM, UDT and vine, as the median p values are greater than the significance level. The name of vine comes from an example of a vine-structure hierarchy in Blanchard and Geman (2005). In their work, they focus on pattern filtering in which the sequence of attribute tests is performed. The attribute tests distinguish the filtered patterns ($\hat{\boldsymbol{Y}}$) from background, which represents "no pattern of interest". If all attribute tests respond positively, then $y_* \in \hat{\boldsymbol{Y}}$ is declared and otherwise the outcome is $y_* \notin \hat{\boldsymbol{Y}}$. Our vine concept is slightly different as we do not have "no pattern of interest", and the attribute tests become OVA-based classifiers.

We show the pseudocode of vine in Algorithm 11 compared to the pseudocode of the UDT, which is reproduced from Chapter 2 for easy comparison. The pseudocode for the testing phase is still the same, as shown in Algorithm 2. This vine idea sets the order of classifiers from the root to leaf node by ranking the number of each class data from the maximum (lower rank) to minimum (higher rank) rather than spending a lot of time searching for the optimal representative classifier for each node. In this setting, vine constructs only $K-1$ classifiers and the tuning time becomes $225T(K-1)$ time units compared to $225T(\frac{K}{2}(1+K)-2)$ in the previous UDT example. Using the number of data in each class as the criteria provides the simplest and fastest way to select the classifier order. Starting with the maximum number will most likely obtain less effect from a class imbalance than other methods when transforming into a binary classifier in

TABLE 6.13: Comparison of accuracy rate among full SVM, vine and UDT on the multi-class problems.

dataset	full SVM	vine	UDT
glass	73.70 ± 9.55	72.11 ± 7.65	74.02 ± 8.47
iris	97.33 ± 4.66	96.67 ± 4.71	97.33 ± 4.66
MNIST	96.23 ± 1.27	94.74 ± 1.88	95.69 ± 1.72
satimage	92.62 ± 0.84	92.00 ± 0.90	92.32 ± 1.18
SCOP-CSH	61.59 ± 7.22	56.03 ± 4.77	61.72 ± 5.36
SCOP-CSHPVZ	62.75 ± 6.07	55.89 ± 5.15	60.32 ± 3.94
segment	97.49 ± 1.43	97.36 ± 0.88	97.62 ± 0.90
shuttle	99.92 ± 0.03	99.91 ± 0.03	99.93 ± 0.03
thyroid	97.73 ± 4.42	97.73 ± 4.42	97.73 ± 4.42
vowel	99.60 ± 0.52	99.29 ± 0.68	99.80 ± 0.43
waveform	87.18 ± 1.85	86.88 ± 1.93	86.88 ± 1.93
wine	99.44 ± 1.76	98.89 ± 2.34	99.44 ± 1.76

TABLE 6.14: Summary results of the proposed hypothesis testing on full SVM, vine and UDT comparison.

	median p value		
	MC Sampling	MCMC+KDE	Bootstrapping
SVM vs UDT	0.22	0.23	0.22
SVM vs vine	0.14	0.18	0.14
UDT vs vine	0.15	0.19	0.15

each node. Furthermore, at subsequent levels, the data are purified to the true binary class as the number of classes is gradually reduced. The lower amount of required training time results from the fixed order, less number of tuning, and less number of data in the lower levels of the tree; hence, vine spends less computational time than the usual one.

Vine requires $K-1$ classifiers compared to $K(K-1)/2$ classifiers using the OVO scheme, in which K represents the number of classes. Vine's classifiers are based on the OVA scheme but they work faster because the number of instances and classes are reduced for constructing the classifiers due to the characteristics of its tree structure, whereas OVA runs the same number of instances on the K classifiers. However, it still requires a great amount of time if each class has a large number of instances. It is possible to combine LDR1SNED and vine to further speed up the process. Table 6.15 shows the performance of this combination on the multi-class problems. We also adds two more large datasets: `full MNIST` and `NECTEC`. The `full MNIST` is the images of numbers zero to nine, and it includes the same number of features and classes as the `MNIST` dataset, except that it is the full version of the `MNIST` dataset, consisting of 70000 instances. The `NECTEC` dataset is the images of Thai and English characters in which there are 65 features,

Algorithm 11 Pseudocode of vine in the training phase

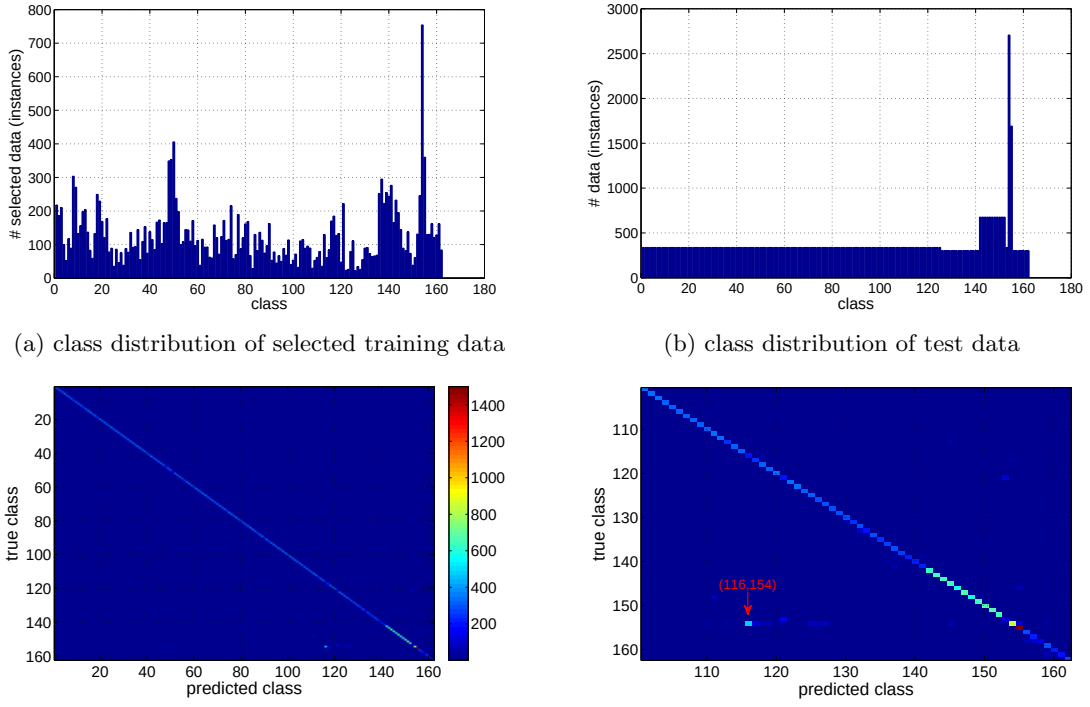
Input: training data (TR), validation data (Val), parameter ranges for tuning ($\boldsymbol{\theta}$)
Output: vine nodes consisting of $(K-1)$ OVA classifiers and $last\_class$

1: vineOrder ← ranking the number of class data from maximum to minimum
2: remove the last element of vineOrder to $last\_class$
3: orgTR ← TR
4: **for** each value i in parameter ranges $\boldsymbol{\theta}$ **do**
5: TR ← orgTR; level = 1
6: **for** each j in vineOrder **do**
7: construct j-vs-All with $\boldsymbol{\theta}_i$
8: nodes(level) = j-vs-All; level = level+1
9: remove data class j from TR
10: **end for**
11: A_i = accuracy from running vine on Val using current nodes
12: **end for**
13: Find $\boldsymbol{\theta}_{opt}$ corresponding to maximum A_i
14: TR ← orgTR; level = 1
15: repeat lines 6-10 with $\boldsymbol{\theta}_{opt}$

Algorithm 12 Pseudocode of UDT in the training phase

Input: training data (TR), validation data (Val), number of classes ($K > 2$), parameter ranges for tuning ($\boldsymbol{\theta}$)
Output: UDT nodes consisting of $(K-1)$ OVA classifiers and $last\_class$

1: $k = \{1, \ldots, K\}$, level = 1
2: **while** $K > 2$ **do**
3: **for** each class j remaining in k **do**
4: **for** each value i in parameter ranges $\boldsymbol{\theta}$ **do** {loop for tuning parameters}
5: get accuracy A_i by running classifier j-vs-All with $\boldsymbol{\theta}_i$ on Val
6: **end for**
7: Find $\boldsymbol{\theta}_{opt}$ corresponding to maximum A_i
8: $score(j)$ = accuracy from running classifier j-vs-All with $\boldsymbol{\theta}_{opt}$ on Val
9: **end for**
10: Find j-vs-All with the maximum $score$
11: nodes(level) = j-vs-All
12: level = level+1; $k = k - \{j\}$; $K = K - 1$
13: remove data of class j from TR, Val and reset $score$ to zero
14: **end while**
15: {at this step TR have only two classes; randomly pick one as j}
16: nodes(level) = j-vs-All
17: $k = k - \{j\}$; $last\_class = k$
18: construct classifier j-vs-All where its parameters are tuned as same as lines 4-7



(a) class distribution of selected training data

(b) class distribution of test data

(c) confusion matrix in colour map

(d) zoom-in of (c) for class 100th – 162th

FIGURE 6.8: Confusion matrix and class ditributions of `NECTEC` dataset.

162 classes and 614376 instances. It is not surprising that the hybrid scheme reveals a similar performance as the LDR1SNED because the latter is run on top of vine. It still produces a lower accuracy rate than the full SVM. However, for the large problems, it also produces an approximate 90 percent accuracy rate, whereas it runs on much smaller training data. This means that it runs faster than the full SVM. For example, only 3794 and 20689 data were used by the hybrid rather than 63000 and 552940 data by the full SVM on the `full MNIST` and `NECTEC` dataset, respectively. The reduction rates on these datasets are 93.98 percent and 96.26 percent, respectively.

Table 6.15: Summary results of the combination of LDR1SNED and vine on multi-class problems compared to SVM and LDR1SNED.

dataset	full SVM	LDR1SNED	LDR1SNED+Vine
glass	73.70 ± 9.55	63.22 ± 10.14	63.14 ± 9.32
iris	97.33 ± 4.66	96.67 ± 4.71	95.33 ± 7.06
MNIST	96.23 ± 1.27	91.34 ± 1.30	88.63 ± 1.67
satimage	92.62 ± 0.84	83.76 ± 1.38	83.85 ± 1.46
SCOP-CSH	61.59 ± 7.22	43.13 ± 4.30	39.98 ± 2.91
SCOP-CSHPVZ	62.75 ± 6.07	48.56 ± 8.86	42.28 ± 8.73
Continued on Next Page...			

Table 6.15 – Continued

dataset	full SVM	LDR1SNED	LDR1SNED+Vine
segment	97.49 ± 1.43	91.99 ± 1.25	90.95 ± 2.35
shuttle	99.92 ± 0.03	93.70 ± 0.82	94.94 ± 0.90
thyroid	97.73 ± 4.42	96.82 ± 6.08	97.73 ± 5.77
vowel	99.60 ± 0.52	71.31 ± 5.84	72.22 ± 5.79
waveform	87.18 ± 1.85	87.12 ± 1.67	86.82 ± 1.56
wine	99.44 ± 1.76	98.89 ± 2.34	98.33 ± 2.68
Full MNIST	98.71 ± 0.12	95.35 ± 0.33	94.44 ± 0.42
NECTEC	97.32 ± 0.06	90.92 ± 0.13	88.73 ± 0.14

TABLE 6.16: Summary results of the proposed hypothesis testing on full SVM, LDR1SNED and LDR1SNED+vine comparison.

	median p value		
	MC Sampling	MCMC+KDE	Bootstrapping
SVM vs LDR1SNED	0.0012	0.0027	0.0012
SVM vs LDR1SNED+vine	4.05×10^{-4}	7.51×10^{-4}	4.65×10^{-4}
LDR1SNED vs LDR1SNED+vine	0.19	0.20	0.20

It is seen that the combination of LDR1SNED and vine produces a lower accuracy rate than the others for the `NECTEC` dataset. In response to this case, we then plot the confusion matrix and class distributions in Figure 6.8, which shows that the 154th class has the maximum number of data. Therefore, this class is used at the root node of the vine as the positive class ($\approx 4\%$), and the other classes behave as the negative class ($\approx 96\%$). The number of test data in the 154th class is 2707 instances, and the vine correctly predicts this class of 848 instances or approximately 31 percent. In other words, the false negative rate of the 154th class is approximately 69 percent, which is quite high. Furthermore, Figure 6.8(d) reveals that the vine incorrectly predicted the 154th class as the 116th class (shown as a cyan pixel in the image at coordinate (116,154)), which is the maximum incorrect prediction (479 instances) approximately one-quater of the false negatives of this true class. It is expected that the error concerning the 154th class is the main factor of the lower performance, which may be caused by imbalanced data or the difficulty of this class itself.

6.5 Evaluation of the Tanimoto Kernel

Much of the running time on the training step involves tuning the trade-off between training error and margin, that is, the C parameter and the kernel parameter such

as γ in the RBF kernel. Normally, we search for the optimal of these parameters on the pre-defined grid of parameter values. For example, C is searched on the range of $\{2^{-2}, 2^{-1}, \ldots, 2^{12}\}$ and γ is searched on the range of $\{2^{-10}, 2^{-9}, \ldots, 2^{4}\}$. Therefore, there are 225 possible combinations of these parameters, which require a longer tuning time, especially on large datasets. The linear kernel is a free parameter kernel, that is, only tuning C. Hence, it makes the tuning time faster than the RBF kernel. In some problems such as bioinformatics and chemoinformatics which have a large number of dimensions or instances, the free parameter kernel is preferable because of the running speed. One of the free parameter kernels is the Tanimoto kernel, which has been effectively applied to the bioinformatics and chemoinformatics domain. It originates from the "Tanimoto coefficient" (Tanimoto, 1958) which is a measure of similarity between two binary strings in the field of chemoinformatics. The higher degree of similarity between the two binary strings, the coefficient is closer to one. In other words, the Tanimoto coefficient is one when the two binary strings are the same, and it is zero when they are completely different. The Tanimoto coefficient is calculated by

$$\text{Tanimoto coefficient} = \frac{c}{a + b - c} \tag{6.2}$$

where a is the number of bit '1' in the first string and b is the number of bit '1' in the second string, whereas c is the number of the common bit '1' in both strings. The calculation of the Tanimoto coefficient is simple and fast, as only the number of bit '1' is counted in the binary strings. We also tried the Tanimoto coefficient in the ECOC and hoping that it would perform better than the Hamming distance. However, it was disappointing to note that there is no improvement by applying the Tanimoto coefficient. Subsequently, the Tanimoto coefficient is used in the SVM as a kernel and referred to as the Tanimoto kernel, which can be similarly calculated by using the following equation:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle}{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle + \langle \boldsymbol{x}_j, \boldsymbol{x}_j \rangle - \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle} \tag{6.3}$$

where $\langle a, b \rangle$ is the inner product between a and b. Tuna and Niranjan (2009); Trotter (2006); Trotter and Holden (2003) claim that the Tanimoto kernel performs better than the linear kernel on binary data. However, we argue that both the Tanimoto and linear kernels are generally comparable. Here, using synthetic data we perform a critical evaluation to explore whether there is indeed an advantage of using the Tanimoto kernel, and if so, what aspects of the binary space may contribute to this advantage. We utilise the following procedure in which the number of dimensions, number of positive instances and number of negative instances are varied:

1. Random positive mean (M_1) and negative mean (M_2) in binary bits of length D where D represents the number of dimensions.

2. Construct $N_1 \times D$ and $N_2 \times D$ matrices where N_1, $N_2 = 2 \times (100 - A)N_1$ are the number of positive and negative instances, respectively. A is the percentage of

positive class training. Each element in the matrices is a random number between 0 and 1, which represents its probability of the corresponding bit.

3. If the probability of each bit in the above step is less than or equal to the probability of the flipping bit, which varies between 0 and 0.5, in order to consider which bit in the means should be flipped, the corresponding bit in the matrices is set to bit '1' and otherwise bit '0'. The binary-bit matrices of this operation can be called F_1 and F_2, where F_1 is the binary version of the $N_1 \times D$ matrix and F_2 is the binary version of the $N_2 \times D$ matrix.

4. Construct binary data of each class by $XOR(M_i, F_i)$ where XOR is the exclusive-or operation between M_i and F_i, and $i = 1, 2$. The binary data is then shuffled and split several times into two portions as training and test data in order to produce multiple sets of training-test pairs.

5. To arrange the above training-test pairs, the positive instances N_1 are divided into $A\,N_1$ and $(100 - A)\,N_1$, and the negative instances N_2 are divided in half where each half has $(100 - A)\,N_1$ data points of the negative class. Then the portions of the $A\,N_1$ positive class and $(100 - A)\,N_1$ negative class data are set as training data and the remaining data as test data. The value of A is set to 50 percent for the balanced class and set to a smaller value such as 20 percent for the imbalanced class.

6. Apply the Tanimoto and linear kernels for each training-test pair and then average the results measured by the area under the ROC curve (AUC). This produces two average AUCs: $AUC_{Tanimoto}$ and AUC_{linear} associated with the means in step 1.

7. Repeat all of the above steps several times and then average all of the corresponding AUC and plot the total average $AUC_{Tanimoto/linear}$ against the flipping-bit probability.

We show the graph between the average AUC and the flipping-bit probability on the balanced and imbalanced class in Figure 6.9 and 6.10, respectively. On the left column, the number of dimensions is 100, whereas the number of the positive class is 100, 500 and 1000 from the top to the last row. The right column graphs are similar to the left column graphs, but the number of dimensions is 1000. The fraction A of positive instances are reserved for the positive training to force the data into becoming an imbalanced class, as described above. The figures show that the AUCs of the Tanimoto and linear kernel are very similar. The difference in both AUCs occurs at the third or later precision after the decimal place, which is difficult to clearly see from the figures. This means that the Tanimoto kernel performs equivalently to the linear kernel on the synthetic data. The flipping-bit probability (p) acts as the noise level perturbing to each mean class, and the figures also show that the larger the number of dimensions, the more robust the perturbation, that is, the larger value of p before the curves start falling.

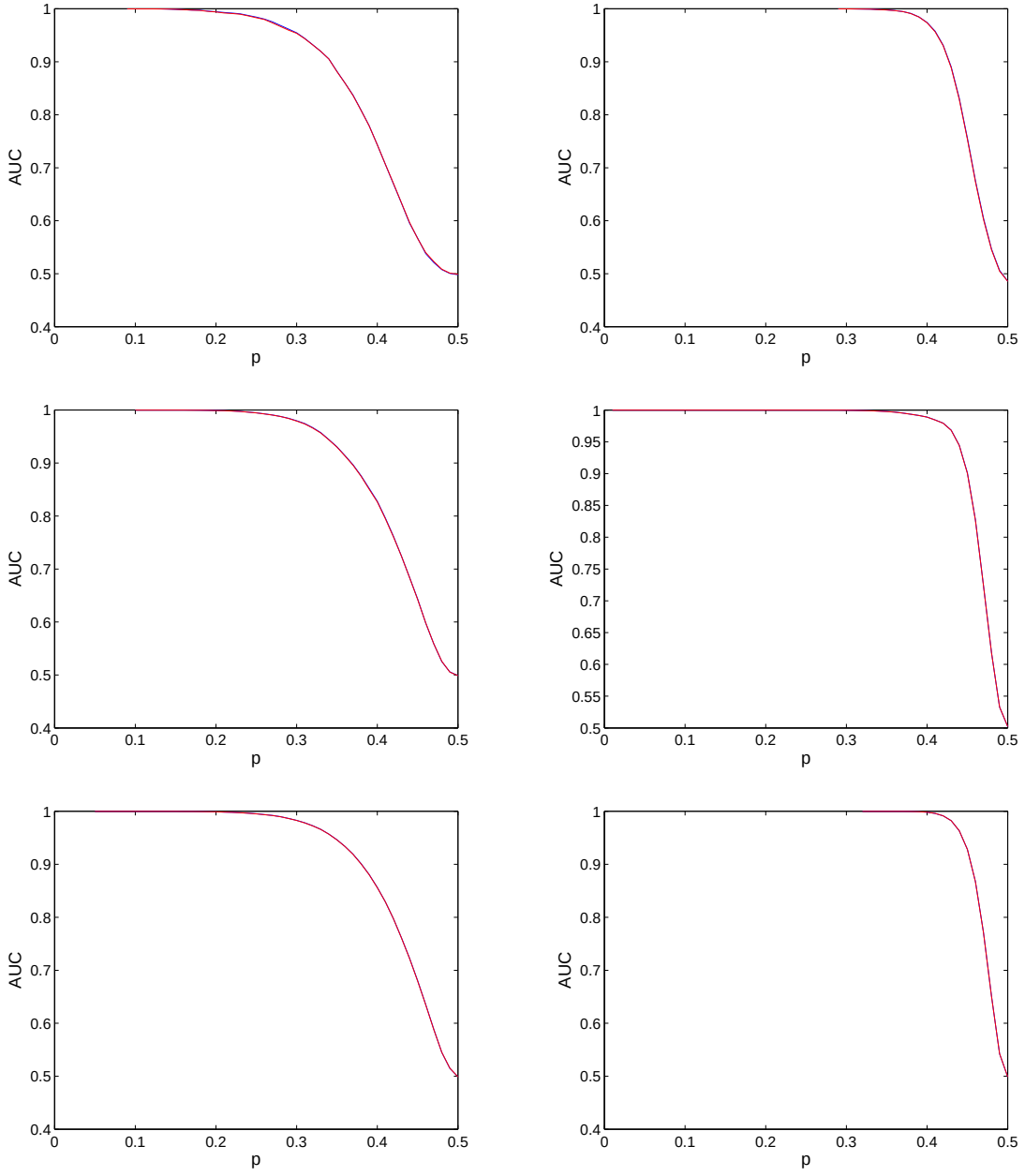


FIGURE 6.9: Graphs between average AUC and the flipping probability, p, on the balanced class by varying the number of instances and dimensions. The number of dimensions is fixed at 100 on the left column and 1000 on the right column, whereas the number of positive instances is fixed at 100, 500 and 1000 from the top to the bottom rows, respectively. The red line represents the AUC of the Tanimoto kernel and the blue line represents the AUC of the linear kernel.

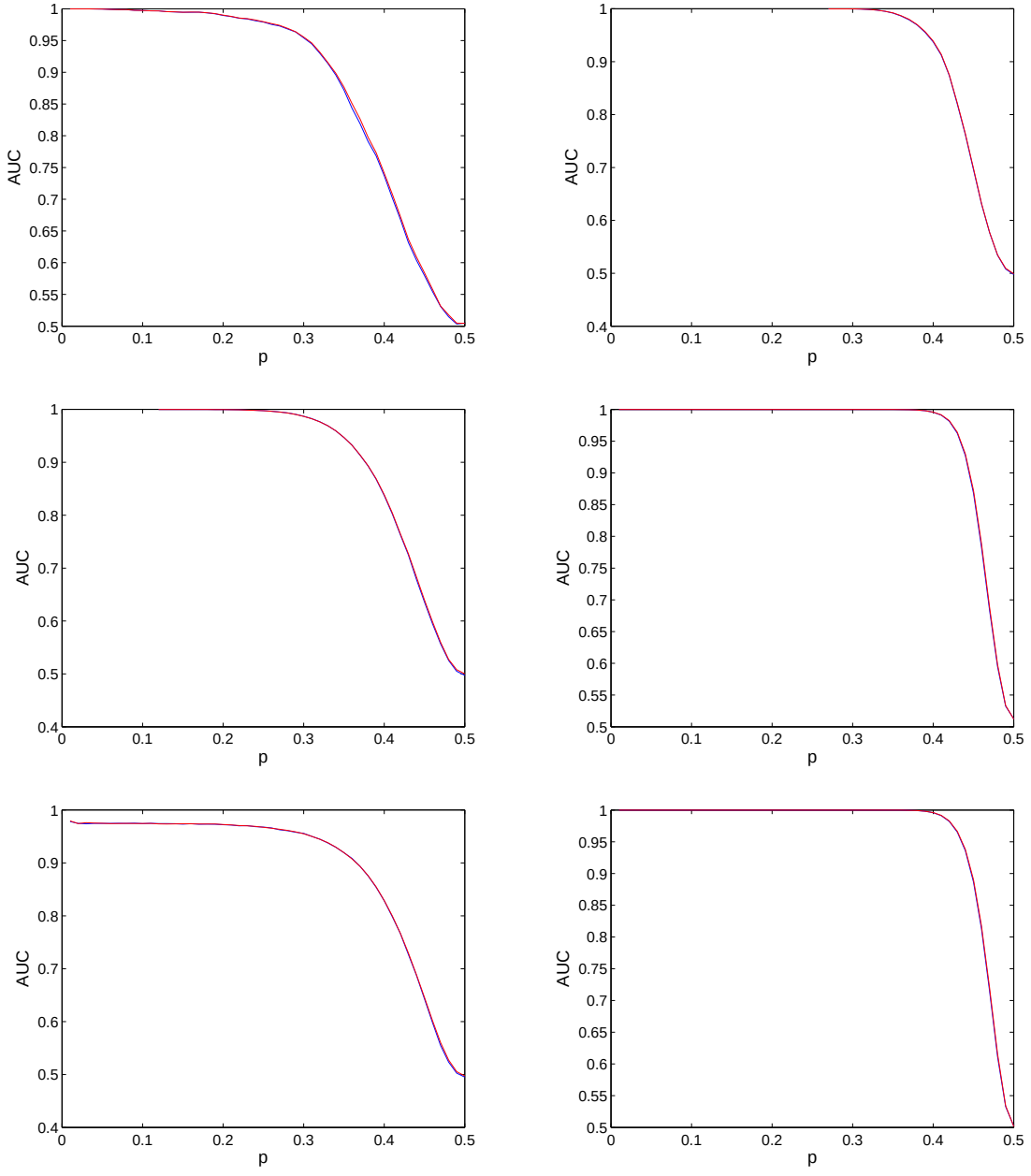


FIGURE 6.10: Graphs between average AUC and the flipping probability, p, on the imbalanced class (20:80) by varying the number of instances and dimensions. The number of dimensions is fixed at 100 on the left column and 1000 on the right column, whereas the number of positive instances is fixed at 100, 500 and 1000 from the top to the bottom rows, respectively. The red line indicates the AUC of the Tanimoto kernel and the blue line represents the AUC of the linear kernel.

A number of bioinformatics and chemoinformatics datasets have also been used to show the SVM's performance using Tanimoto, linear and RBF kernels, as well as compared to GP. Trotter (2006) suggested that using the Tanimoto kernel on chemoinformatics improves the performance. However, the following experiments argue that even though the Tanimoto kernel may improve performance, it is comparable to the performance of other kernels. The chemoinformatics dataset is the `chemo` dataset shown in Table 6.1. The bioinformatics datasets shown in Table 6.17 have a small number of instances compared to their dimensions. Hence, we randomly split in half 100 times for the training and test data on each dataset. These training and test data are called raw training and raw test data, respectively. To binarise data (transform raw data into binary data, i.e. '0' or '1') in each dimension, we use the mean of the corresponding dimension of the raw training data as a threshold. The thresholds from the raw training data are also used for binarising the raw test data. The raw data are set to bit '1' if their values are greater than or equal to the thresholds, and otherwise bit '0'. The test data of both the raw and binary versions are then fed to the classifiers that are constructed from the corresponding training data, and the area under the ROC curves are measured. In this experiment, the classifier SVM is used with 1) linear kernel, 2) RBF kernel, 3) Tanimoto kernel and classifier GP with a squared exponential covariance function and cumulative Gaussian likelihood are used. Table 6.18 shows the area under the ROC curve where 'RAW' and 'BIN' indicate that the classifiers are fed the raw and binary data, respectively. On the contrary, the chemoinformatics dataset is already in the binary form due to the fingerprint process which is commonly used in its domain. Therefore, there are no results reported on the 'RAW' panel. The dataset is run via 10CV, as it contains sufficient data.

Table 6.17: Summary characteristics of the bioinformatics datasets

dataset	# positive	# negative	# dimension
colon	22	40	2000
lymphnode	10	10	22283
prostrate	69	20	22277

Given that the number of dimensions or features of the datasets is large, one might be tempted to reduce it. Table 6.19 shows the new results when we apply the feature selection based on the Fisher ratio. The number of selected features is roughly calculated by the square root of the number of original features. It turns out that the feature selection improves performance. According to the results shown in Table 6.18–6.19, we can see that the Tanimoto kernel performs comparably to other kernels. Table 6.20 shows the median p values using the Monte Carlo sampling method, in which the Tanimoto kernel is used as the base classifier. The results support that there is no significant difference between Tanimoto and other kernels in this particular domain. The median p values using MCMC with KDE and bootstrapping methods are also approximately

TABLE 6.18: AUC comparison between SVM with linear, RBF, and Tanimoto kernels and on GP with the raw data (top) and binary data (bottom).

		colon	lymphnode	prostrate	chemoinformatics
RAW	linear	0.85 ± 0.06	0.99 ± 0.05	0.89 ± 0.06	-
	RBF	0.82 ± 0.08	0.98 ± 0.06	0.86 ± 0.08	-
	GP	0.86 ± 0.06	0.99 ± 0.03	0.89 ± 0.07	-
	Tanimoto	0.88 ± 0.05	0.98 ± 0.06	0.89 ± 0.06	-
		colon	lymphnode	prostrate	chemoinformatics
BIN	linear	0.82 ± 0.07	0.99 ± 0.05	0.91 ± 0.05	0.97 ± 0.006
	RBF	0.81 ± 0.07	0.99 ± 0.06	0.91 ± 0.05	0.99 ± 0.004
	GP	0.82 ± 0.07	0.99 ± 0.05	0.90 ± 0.06	0.99 ± 0.005
	Tanimoto	0.83 ± 0.06	0.98 ± 0.06	0.91 ± 0.06	0.99 ± 0.004

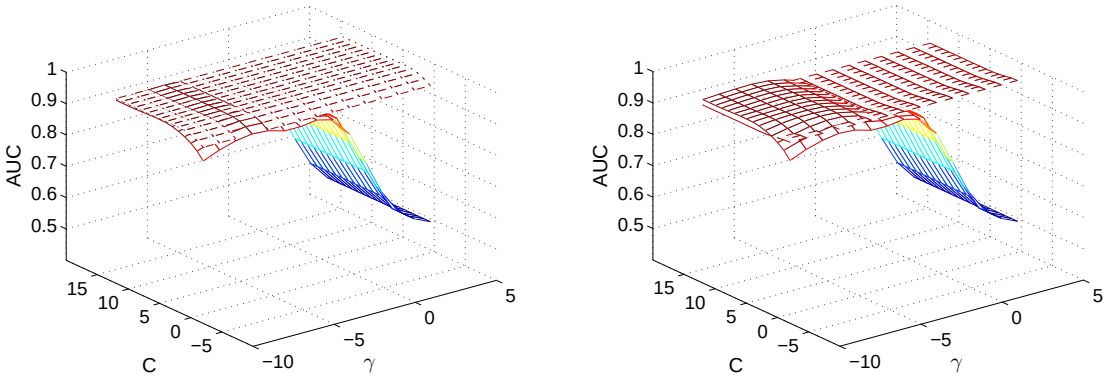
TABLE 6.19: AUC comparison between SVM with linear, RBF, and Tanimoto kernels and on GP with the raw data (top) and binary data (bottom) after feature selection.

		colon	lymphnode	prostrate	chemoinformatics
RAW	linear	0.92 ± 0.04	1.00 ± 0.00	0.92 ± 0.04	-
	RBF	0.94 ± 0.03	1.00 ± 0.00	0.95 ± 0.03	-
	GP	0.93 ± 0.04	1.00 ± 0.00	0.95 ± 0.03	-
	Tanimoto	0.93 ± 0.04	1.00 ± 0.00	0.92 ± 0.04	-
		colon	lymphnode	prostrate	chemoinformatics
BIN	linear	0.89 ± 0.05	1.00 ± 0.00	0.93 ± 0.04	0.93 ± 0.009
	RBF	0.91 ± 0.05	1.00 ± 0.00	0.94 ± 0.03	0.95 ± 0.007
	GP	0.91 ± 0.05	1.00 ± 0.00	0.94 ± 0.03	0.96 ± 0.007
	Tanimoto	0.91 ± 0.04	1.00 ± 0.00	0.93 ± 0.04	0.95 ± 0.01

TABLE 6.20: Summary of median p values using Monte Carlo sampling on GP and SVM with linear, RBF and Tanimoto kernels. SVM with Tanimoto kernel is used as the base classifier in this comparison.

	without feature selection	with feature selection
Tanimoto vs linear	0.22	0.20
Tanimoto vs RBF	0.19	0.19
Tanimoto vs GP	0.22	0.21

0.2; hence, they are not shown in the separated table. In comparison between the raw and binary data, most of the classifiers fed raw data have a higher AUC than the classifiers fed binary data, although working with binary data is faster. However, there is a disadvantage of working with binary data: the data should have long bits, as the shorter bit length, the higher probability that all bits are equal to one or zero. It is also possible that for the shorter bit length, the same binary strings fall into a different class.



(a) RBF kernel vs Tanimoto kernel

(b) RBF kernel vs linear kernel

FIGURE 6.11: Distribution of AUC when the SVM parameters are varied in the power of two ,i.e. 2^C and 2^γ. (a) AUC distribution when varying the parameters between the Tanimoto kernel (dashed hyperplane) and RBF kernel (solid hyperplane). (b) AUC distribution when varying the parameters between the linear kernel (dashed hyperplane) and RBF kernel (solid hyperplane).

The results support the suggestion to use the Tanimoto kernel on the chemoinformatics domain because of the nature of the problem, which has very large dimensions and does not lose much information when binarising the data. Nonetheless, it is possible that binarising data may result in the loss of some useful information, as seen in the `colon` dataset in which the performance level drops between the raw and binary version. Another reason why the Tanimoto kernel is preferable in chemoinformatics is that it is not very sensitive to the C parameter in SVM. As shown in Figure 6.11(a), the AUC distribution of the Tanimoto kernel denoted by a dashed hyperplane is quite stable, even though the trade-off parameter C values are tuned. On the contrary, the AUC distribution of the RBF kernel denoted by a solid hyperplane is affected when the parameters are tuned. This means that the Tanimoto kernel does not require a wider range of tuning parameter values. Figure 6.11(b) shows a similar graph, but with plotting on the linear kernel (dashed hyperplane) and RBF kernel (solid hyperplane). Note that the Tanimoto and linear kernels do not have a kernel parameter γ; therefore, they are not affected by the γ.

6.6 Summary

We have proposed a number of algorithms as a pre-process method of reducing the number of instance inputs before running the SVM in order to alleviate the demand of memory and computational running time during the training step. The proposed methods were compared to the SVM trained with the entire training data, referred to as full SVM. Among the proposed data selection methods, merging the APC with

Almeida's approach and the thresholded Gaussian method performed comparably to the full SVM, even though their data reduction rates are low. LDR offers a medium reduction rate, while its performance does not degrade very much compared to the full SVM. LDR1SNED is the method that offers the highest data reduction rate, even though the accuracy rate is lower than the full SVM. The accuracy rate can be increased by modifying the LDR1SNED, but the trade-off is a longer pre-processing time. LDR1SNED is a suitable method when the dataset consists of a very large number of instances and cannot be loaded into memory. When there are a large number of classes, we have discussed an improved technique of UDT called vine, which speeds up the training time. In the case of a very large number of instances and classes, running LDR1SNED on top of vine can be used to speed up the training. However, the performance could degrade as a result of losing too many qualitative data. In such cases, LDR1S2NED should be used before running vine to obtain better performance. To further speed up the training, a free parameter kernel such as the Tanimoto kernel can be used instead of the RBF kernel, although the performance may be lower because of the binarisation. We have shown that the performance of the Tanimoto kernel is not significantly different from that of the linear and other kernels in the binary data. However, the Tanimoto kernel performs faster and its performance does not vary so much though the trade-off parameter is tuned in a wide range.

Chapter 7

Conclusions

This thesis documents a systematic empirical comparison on a number of high-performance pattern classification methods. Support vector machines and Gaussian processes are the particular focus of our attention, due to their popularity of usage in recent machine learning literature. However, we have shown that it is difficult to reach a concrete conclusion in the literature as a result of inconsistency in the experimental settings and data usage. We made an attempt to use available statistical tests to compare the results in the literature, but found that most of the authors omitted the sufficient information and we could not use suitable tests. Even though one may provide such information, perhaps the statistical hypothesis test is not appropriate in the classification problem when running the cross-validation or several split training-test data pairs to avoid the bias of using only a specific fixed training-test dataset. Hence, we have developed the hypothesis testing that considers the uncertainty due to the cross-validation which the statistical tests and most researchers fail to use. The main contribution of this work is to make use of the uncertainty integrated into the hypothesis testing for comparison of classifiers. It was evident that our hypothesis testing is an appropriate test, and it is more appropriate than the recommended non-parametric hypothesis testing under circumstances detailed in this thesis.

The scalability problem of Gaussian processes can be alleviated by using the active set in sparse GP approximation, whereas we have support vectors, which is the sparse solution, for SVM. However, support vectors are obtained by solving the quadratic programming, which also has the scalability issue. We made a contribution towards the sparse data for the full SVM by using data selection approaches. The aim is to select only a subset of potential data in which they are likely to be candidate support vectors. This reduces the computational time, as the QP only focuses on a small number of informative data. With the data selection, we also developed a new learning tree structure designed for the classification problems with a large number of classes and instances. Recently, the Tanimoto kernel was applied to binary data and it was claimed that it is better than other kernels, especially in chemoinformatics. We showed that the Tanimoto kernel is

comparable to other kernels. Although it may improve the performance, the performance difference is not statistically significant. The thesis's contribution and limitations, as well as the future work, are summarised in this chapter.

7.1 Summary of Thesis

The main contribution and work done in this thesis include:

- Introducing new sampling-based methods of statistical comparison of classifiers, considering the uncertainty in the 10CV. The more complex method is based on Markov chain Monte Carlo sampling, which requires longer computational time due to the drawing samples from the target distribution. The simpler methods are based on Monte Carlo sampling and bootstrapping. These methods resample the uncertainty from the ten-fold cross-validation, compute the median p value, and then make a decision concerning the significance level. The hypothesis testing with uncertainty is the main contribution of this thesis, and it was used later in the work and its application done below.

- Proposing data selection to enhance the off-line SVM. Given that only a subset of training data, known as support vectors, is required for the SVM to construct the optimal hyperplane, the proposed methods filter out many redundant data that are not likely to be support vectors. Many training data are eliminated making it possible to run the SVM on a machine with limited memory when all of the training data are too large to load into memory. LDR1SNED is a proposed method for the pre-process operation for the SVM, which offers the maximum reduction rate compared to other methods. LDR1S2NED should be used when a lot of qualitative data are filtered out.

- Improving a new SVM-based tree structural classifier called a vine, which is an accelerated version of the UDT for a large number of classes. Further fast training can be provided via the hybridisation of the LDR1SNED or LDR1S2NED and vine in the case of a large number of instances and classes.

- Demonstrating and arguing that SVM with the Tanimoto kernel performs comparably to other kernels on binarised data but works faster and is more robust on the parameter tuning.

7.2 Limitations

The proposed methods, both statistical hypothesis testing with uncertainty in Chapter 5 and data selection in Chapter 6, have made certain assumptions. Hence, the following limitations exist:

- The assumption that the test data must be drawn from the same or similar distribution of the training data. This is a normal assumption in the classification problem; however, it is not always true in many problems. The performance of the proposed data selection methods may degrade if the test data and the selected training data distributions are very different.
- The family of LDRs assumes that all data features can be loaded into the computer memory. In the case of problems with a very large number of dimensions, the LDRs must resort to a feature selection until $N > D\sqrt{D}$.
- In the proposed statistical hypothesis testing with uncertainty, the median p value is used as an ad hoc threshold. However, other thresholds may be used and investigated.
- As the proposed hypothesis testing with uncertainty is based on sampling from the 10CV, the approximation of the performance distribution on these ten values may not be the most accurate.

7.3 Future Work

A few large datasets are used in the experiments for demonstrating and comparing the possibility of the proposed data selection methods with the full SVM due to the high demand in training time, particularly in parameter tuning, and the limited resources such as limited running job time and memory usage. With additional resources, more datasets with a large number of classes and instances should be used to measure the potential of the proposed data selection methods against the full SVM. Another possible way to avoid memory shortage is using an online SVM although it still requires a long running time due to a large number of instances and parameter tuning. It is worth comparing the online SVM with the proposed methods or applying a combination of the proposed methods with the online SVM to determine the effect. In the case of a very large problem, we expect that splitting the problem into many smaller subproblems and then running the proposed data selection methods and collecting the selected data as the selected training will simultaneously speed up and reduce the memory problem. Kernelising the proposed methods would pave another avenue for future work. It might be of interest to examine the effect of the proposed methods on the full GP and compared it to the full SVM. Many derivations of the proposed methods could be used to determine

if they have any merit. Different binarisation schemes for the Tanimoto kernel are also of interest to determine the effect of the binarisations on performance.

Bibliography

S. Abe. Training of Support Vector Machines with Mahalanobis kernels. In *Artificial Neural Networks: Formal Models and Their Application*, volume 3697 of *Lecture Notes in Computer Science*, pages 571–576. Springer Berlin/Heidelberg, 2005.

D. W. Aha, D. Kibler, and M. Albert. Instance-Based Learning Algorithms. *Machine Learning*, 6:37–66, 1991.

F. Aiolli and A. Sperduti. Multiclass Classification with Multi-Prototype Support Vector Machines. *Journal of Machine Learning Research*, 6:817–850, 2005.

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

M. B. Almeida, A. P. Braga, and J. P. Braga. SVM-KM: speeding SVMs learning with a *priori* cluster selection and k-means. In *Proceedings of the 6th Brazilian Symposium on Neural Networks*, pages 162–1167, 2000.

A. Asuncion and D.J. Newman. UCI Machine Learning Repository. Available electronically at `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2007.

A. M. Bagirov, A. M. Rubinov, N. V. Soukhoroukova, and J. Yearwood. Unsupervised and Supervised Data Classification Via Nonsmooth and Global Optimization. *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research*, 11 (1):1–75, 2003.

D. Bamber. The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. In *Journal of Mathematical Psychology*, volume 12, pages 387–415, 1975.

D. Barber and C.K. Williams. Gaussian process for bayesian classification via hybrid monte carlo. pages 340–346, 1997.

K.P. Bennett and C. Campbell. Support Vector Machines: Hype or Hallelujah? In *SIGKDD Explorations*, volume 2, pages 1–13, 2000.

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

G. Blanchard and D. Geman. Hierarchical Testing Designs for Pattern Recognition. *The Annals of Statistics*, 33(3):1155–1202, 2005.

D. Boley and D. Cao. Training Support Vector Machine using Adaptive Clustering. In *Proceedings of the 4th SIAM International Conference on Data Mining*, pages 126–137, 2004.

H. Byun and S.W. Lee. A Survey on Pattern Recognition Applications of Support Vector Machines. In *International Journal of Pattern Recognition and Artificial Intelligence*, volume 17, pages 459–486, 2003.

J. Cervantes, X. Li, W. Yu, and K. Li. Support Vector Machine Classification for Large Data Sets via Minimum Enclosing Ball Clustering. *Neurocomputing*, 71(4-6):611–619, 2008.

C. Cortes and V. Vapnik. Support-Vector Networks. In *Machine Learning*, volume 20, pages 273–297, 1995.

N. Cristianini and J. Shawe-Taylor. An introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000.

L. Csató and M. Opper. Sparse Representation for Gaussian Process Models. In T. G. Dietterich T. K. Leen and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 444–450. MIT Press, Cambridge, MA, 2001.

R. Debnath, N. Takahide, and H. Takahashi. A decision based one-against-one method for multi-class support vector machine. In *Pattern Analysis and Application*, volume 7, pages 164–175, 2004.

J. Demšar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

T. G. Dietterich and G. Bakiri. Error-Correcting Output Code: A General Method for Improving Multiclass Inductive Learning Programs. In *Proceedings of the Ninth AAAI National Conference on Artificial Intelligence*, pages 572–577, 1991.

T. G. Dietterich and G. Bakiri. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

C. H. Ding and I. Dubchak. Multi-class protein fold recognition using support vector machines and neural networks. *Bioinfomatics*, 17(4):349–358, 2001.

J. X. Dong. *Speed and accuracy: large-scale machine learning algorithms and their applications*. PhD thesis, Department of Computer Science, Concordia University, Montreal, 2003.

W. Duch. Benchmark datasets used for classification: comparison of results. Available electronically at `http://www.is.umk.pl/projects/datasets.html#Sonar`, 2002.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley & Sons, 2001.

C. W. Dunnett. A Multiple Comparison Procedure for Comparing Several Treatments with a Control. *Journal of American Statistical Association*, 50:1096–1121, 1980.

T. Fawcett. ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Technical Report HPL-2003-4, Hewlett-Packard Laboratories, 2003.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

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

R. A. Fisher. *Statistical Methods for Research Workers.* Oliver and Boyd, 1928.

R. A. Fisher. *Design of Experiments.* London: Oliver & Boyd, 1935.

R.A. Fisher. Combining Independent Tests of Significance. *American Statistician*, 2(5): 30, 1948.

B. J. Frey and D. Dueck. Clustering by Passing Messages Between Data Points. *Science*, 315:972–976, 2007.

M. Friedman. A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *Annals of Mathematical Statistics*, 11:86–92, 1940.

T. V. Gestel, J. A. K. Suykens, G. Lanckriet, A. Lambrechts, B. De Moor, and J. Vandewalle. Bayesian Framework for Least Squares Support Vector Machine Classifiers, Gaussian Processes, and Kernel Fisher Discriminant Analysis. In *Neural Computation*, volume 14(5), pages 1115–1147, 2002.

T. V. Gestel, J.A.K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, and J.Vandewalle. Benchmarking least square support vector machine classifiers. *Machine Learning*, 54:5–32, 2004.

M. N. Gibbs and D. J. C. MacKay. Variational Gaussian Process Classifiers. In *IEEE Transactions on Neural Networks*, volume 11(6), pages 1458–1464, 2000.

M. Girolami and S. Rogers. Variational Bayesian Multinomial Probit Regression with Gaussian Process Priors. In *Neural Computation*, volume 18(8), pages 1790–1817, 2006.

M. Girolami and M. Zhong. Data Integration for Classification Problems Employing Gaussian Process Priors. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 465–472, 2006.

J. Guo, N. Takahashi, and T. Nishi. A Learning Algorithm for Improving the Classification Speed of Support Vector Machines. In *Proceedings of the European Conference on Circuit Theory and Design*, volume 3, pages 381–384, 2005.

D. J. Hand and R. J. Till. A Simple Generalization of the Area Under the ROC Curve to Multiple Class Classification Problems. *Machine Learning*, 45(2):171–186, 2001.

J. A. Hanley and B. J. McNeil. The Meaning and Use of the Area under Receiver Operating Characteristic (ROC) Curve. In *Radiology*, volume 143, pages 29–36, 1982.

W. K. Hastings. Monte Carlo sampling method using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

C. Hsu and C. Lin. A Comparison of Methods for Multi-class Support Vector Machines. In *IEEE Transactions on Neural Networks*, volume 13(2), pages 415–425, 2002.

T. Joachims. Making Large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press, 1999.

D. Keysers, R. Paredes, H. Ney, and E. Vidal. Combination of Tangent Vectors and Local Representations for Handwritten Digit Recognition. In *Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 538–547. Springer-Verlag, London, UK, 2002.

H. Kim and Z. Ghahramani. Bayesian Gaussian Process Classification with the EM-EP Algorithm. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 28(12), pages 1948–1959, 2006.

H. Kim and H. Park. Data Reduction in Support Vector Machines by a Kernelized Ionic Interaction Model. In *Proceedings of the 4th SIAM International Conference on Data Mining*, pages 507–511, 2004.

R. Koggalage and S. Halgamuge. Reducing the Number of Training Samples for Fast Support Vector Machine Classification. *Neural Information Processing-Letters and Reviews*, 2(3), pages 57–65, 2004.

A. Kolmogorov. "Uber die analytischen Methoden in der Wahrscheinlichkeitsrechnung". *Mathematische Annalen*, 104:415–458, 1931.

M. Kuss and C. E. Rasmussen. Assessing Approximation for Gaussian Process Classification. In B. Schölkopf Y. Weiss and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 699–706. MIT Press, Cambridge, MA, 2006.

M. T. Landi, T. Dracheva, M. Rotunno, J. D. Figueroa, H. Liu, A. Dasgupta, F. E. Mann, J. Fukuoka, A. W. Hames, M.and Bergen, et al. Gene Expression Signature of Cigarette Smoking and Its Role in Lung Adenocarcinoma Development and Survival. *PLoS ONE*, 3(2), 2008.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast Sparse Gaussian Process Methods: The Informative Vector Machine. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 609–616. MIT Press, Cambridge, MA, 2003.

Y.-J. Lee and O. L. Mangasarian. RSVM: Reduced Support Vector Machines. In *Proceedings of the 1st SIAM International Conference on Data Mining*, pages 325–361, 2001.

J. Li, G. Dong, K. Ramamohanarao, and L. Wong. DeEPs: A New Instance-Based Lazy Discovery and Classification System. *Machine Learning*, 54(2):99–124, 2004.

X. Li, L. Wang, and E. Sung. A study of AdaBoost with SVM based weak learners. In *Proceedings of IEEE International Joint Conference on Neural Networks*, volume 1, pages 196–201, 2005.

H.-T. Lin and C.-J. Lin. A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods. In *Technical Report*. National Taiwan University, 2003.

X. Liu, H. Xing, and X. Wang. A Multistage Support Vector Machine. In *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, volume 3, pages 1305–1308, 2003.

H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *Annals of Mathematical Statistics*, 18:50–60, 1947.

F. J. Massey. The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of American Statistical Association*, 46(253):68–78, 1951.

R. L. McCornack. Extended Tables of the Wilcoxon Matched Pair Signed Rank Statistic. *Journal of the American Statistical Association*, 60:864–871, 1965.

N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of American Statistical Association*, 44:335–341, 1949.

S. Mika, G. Rätsch, J. Weston, B. Scholköpf, and K. Müller. Fisher Discriminant Analysis with Kernels. In *Proceedings of IEEE Neural Networks for Signal Processing Workshop*, pages 41–48, 1999.

T. P. Minka. Expectation Propagation for Approximate Bayesian Inference. In J. S. Breese and D. Koller, editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann, 2001.

J. M. Moguerza and A. Muñoz. Support Vector Machines with Applications. *Statistical Science*, 21(3):322–336, 2006.

A. Naish-Guzman and S. Holden. The generalized FITC approximation. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1057–1064. MIT Press, Cambridge, MA, 2008.

H. Nickisch and C. E. Rasmussen. Approximations for Binary Gaussian Process Classification. *Journal of Machine Learning Research*, 9:2035–2078, 2008.

A. O'Hagan. Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society, Series B*, 40(1):1–42, 1978.

M. Opper and O. Winther. Gaussian Processes for Classification: Mean Field Algorithms. *Neural Computation*, 12(11):2655–2684, 2000.

Y.-Y. Ou, C.-Y. Chen, S.-C. Hwang, and Y.-J. Oyang. Expediting Model Selection for Support Vector Machines Based on an Advanced Data Reduction Algorithm. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 786–791, 2003.

Y.-Y. Ou, G.-H. Chen, and Y.-J. Oyang. Expediting Model Selection for Support Vector Machines Based on an Advanced Data Reduction Algorithm. In *PRICAI 2006: Trends in Artificial Intelligence*, volume 4099/2006 of *Lecture Notes in Computer Science*, pages 1017–1021. Springer Berlin/Heidelberg, 2006.

E. Parzen. On Estimation of a Probability Density Function and Mode. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

G. Peterson and H. Barney. Control Methods Used in a Study of the Vowels. In *Journal of Acoustical Society of America*, volume 24, pages 175–184, 1952.

J. Platt, N. Cristianini, and J. Shawe-Taylor. Large Margin DAGs for Multiclass Classification. In *Advances in Neural Information Processing Systems*, volume 12, pages 547–553, 2000.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editor, *Advances in Kernel Methods– Support Vector Learning*, pages 185–208. MIT press, 1999.

K. Polat and S. Güneş. Breast Cancer Diagnostic using Least Square Support Vector Machine. *Digital Signal Processing*, 17(4):694–701, 2007.

K. Polat and S. Güneş and A. Arslan. A cascade learning system for classification of diabetes disease: Generalized Discriminant Analysis and Least Square Support Vector Machine. *Expert Systems with Applications*, 34(1):482–487, 2008.

F. Provost and T. Fawcett. Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. In *Proceedings of the Third international conference on Knowledge Discovery and Data Mining*, pages 43–48, 1997.

A. Ramanan, S. Suppharangsan, and M. Niranjan. Unbalanced Decision Trees for multi-class classification. In *Proceedings of the 2nd IEEE International Conference on Industrial and Information Systems*, pages 291–294, 2007.

C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

N. Rhodes, P. Willett, J. B. Dunbar, and C. Humblet. Bit-String Methods for Selective Compound Acquisition. *Journal of Chemical Information and Computer Sciences*, 40 (2):210–214, 2000.

S. L. Salzberg. On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach. In *Data Mining and Knowledge Discovery*, pages 317–328. Kluwer Academic Publishers, Boston, 1997.

H. Scheffé. A Method for Judging all Contrasts in the Analysis of Variance. *Biometrika*, 40:87–104, 1953.

M. Seeger. Bayesian model selection for Support Vector machines, Gaussian processes and other kernel classifiers. In T. K. Leen S. A. Solla and K. R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 603–609. MIT Press, Cambridge, MA, 2000.

M. Sewak, P. Vaidya, and C. Chan. SVM Approach to Breast Cancer Classification. In *Proceedings of the Second International Multi-Symposiums on Computer and Computational Sciences*, pages 32–37, 2007.

S. S. Shapiro and M. B. Wilk. An Analysis of Variance Test for Normality (Complete Samples). *Biometrika*, 52(3-4):591–611, 1965.

D. J. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures (3rd edition)*. Chapman & Hall/CRC, 2004.

H. J. Shin and S. Z. Cho. Pattern Selection for Support Vector Classifiers. In *Proceedings of the 3rd International Conference on Intelligent Data Engineering and Automated Learning*, pages 469–474, 2002.

Z. Songfeng, L. Xiaofeng, Z. Nanning, and X. Weipu. Unsupervised clustering based reduced support vector machines. In *Proceedings of Acoustics, Speech, and Signal Processing*, volume 2, pages 821–824, 2003.

R. C. Sprinthall. *Basic Statistical Analysis*. Allyn & Bacon, Boston, M.A., 2003.

Student. The Probable Error of a Mean. *Biometrika*, 6(1):1–25, 1908.

J. A. Swets. Measuring the Accuracy of Diagnostic Systems. In *Science*, volume 240, pages 1285–1293, 1988.

T. T. Tanimoto. An Elementary Mathematical Theory of Classification and Prediction. IBM Internal Report, 1958.

M. E. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

Q. L. Tran, K. A. Toh, D. Srinivasan, K. L. Wong, and S. Q. C. Low. An Empirical Comparison of Nine Pattern Classifiers. In *IEEE Transactions on Systems Man and Cybernetics Part B*, pages 1079–1091, 2005.

M. Trotter. *Support Vector Machines for Drug Discovery*. PhD thesis, University College London, UK, 2006.

M. Trotter and S. Holden. Support Vector Machines for ADME Property Classification. *QSAR and Combinatorial Science*, 22(5):533–548, 2003.

J. W. Tukey. Comparing Individual Means in the Analysis of Variance. *Biometrics*, 5: 99–114, 1949.

S. Tuna and M. Niranjan. Classification with binary gene expressions. *Journal of Biomedical Science and Engineering*, 2:390–399, 2009.

C. J. Van Rijsbergen. *Information Retrieval*. Butterworths, London, 2 edition, 1979.

V. N. Vapnik. *Estimation of Dependence Based on Empirical Data*. Springer-Verlag, Berlin, 1982.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

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

X. Wang and Y. Zhong. Statistical Learning Theory and State of the Art in SVM. In *Proceedings of The Second IEEE International Conference on Cognitive Informatics*, pages 55–59, 2003.

A. Watkins and L. Boggess. A new classifier based on resource limited artificial immune systems. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 2, pages 1546–1551, 2002.

F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 3: 80–83, 1945.

C. K. I. Williams and D. Barber. Bayesian Classification with Gaussian Processes. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 20(12), pages 1342–1351, 1998.

Z. Xiangrong and L. Fang. A pattern classification method based on GA and SVM. In *6th International Conference on Signal Processing*, volume 1, pages 110–113, 2002.

Z. Zhou and Y. Jiang. NeC4.5: neural ensemble based C4.5. In *IEEE Transactions on Knowledge and Data Engineering*, volume 16(6), pages 770–773, 2004.