University College London

Doctoral Thesis

Optimisation Approaches for Data Mining in Biological Systems

Lingjian Yang

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Centre for Process Systems Engineering
Department of Chemical Engineering

December 2015

Declaration of Authorship

I, Lingjian YANG, declare that this thesis titled, 'Optimisation Approaches for Data Mining in Biological Systems' 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.

Signed: _____

Date: _____

Abstract

The advances in data acquisition technologies have generated massive amounts of data that present considerable challenge for analysis. How to efficiently and automatically mine through the data and extract the maximum value by identifying the hidden patterns is an active research area, called data mining. This thesis tackles several problems in data mining, including data classification, regression analysis and community detection in complex networks, with considerable applications in various biological systems.

First, the problem of data classification is investigated. An existing classifier has been adopted from literature and two novel solution procedures have been proposed, which are shown to improve the predictive accuracy of the original method and significantly reduce the computational time.

Disease classification using high throughput genomic data is also addressed. To tackle the problem of analysing large number of genes against small number of samples, a new approach of incorporating extra biological knowledge and constructing higher level composite features for classification has been proposed. A novel model has been introduced to optimise the construction of composite features.

Subsequently, regression analysis is considered where two piece-wise linear regression methods have been presented. The first method partitions one feature into multiple complementary intervals and fits each with a distinct linear function. The other method is a more generalised variant of the previous one and performs recursive binary partitioning that permits partitioning of multiple features.

Lastly, community detection in complex networks is investigated where a new optimisation framework is introduced to identify the modular structure hidden in

directed networks via optimisation of modularity. A non-linear model is firstly proposed before its linearised variant is presented. The optimisation framework consists of two major steps, including solving the non-linear model to identify a coarse initial partition and a second step of solving repeatedly the linearised models to refine the network partition.

Acknowledgements

I would like to express my gratitude towards my supervisors Prof. Lazaros Papageorgiou and Dr. Sophia Tsoka for offering me invaluable support, guidance and trust throughout my Ph.D study. Many thanks go to Dr. Wolfram Wiesemann at Imperial College, Prof. Alberto Striolo, Prof. David Bogle and my supervisors for various teaching and research opportunities made available to me over the last 4 years.

Financial support from EPSRC Centre for Innovative Manufacturing in Emergent Macromolecular Therapies and Centre for Process Systems Engineering (CPSE) at Imperial College London and University College London are gratefully acknowledged.

I am very thankful for all my colleagues in UCL and KCL, especially Dr. Songsong Liu, Dr. Laura Bennett and Dr. Chrysanthi Ainali who have provided a lot of experience, help and friendship.

Finally, I would love to thank all my dear friends, most of who are Ph.D candidates themselves. It is a pleasure to share those memory with you.

Contents

List of Figures

List of Tables

Dedicated to my parents.

Chapter 1

A General Introduction

The amount of data being generated nowadays has been growing exponentially. Thanks to the advances in data acquisition and digital storage, it is estimated that the world's database doubles its size every 20 months, providing unprecedented resources. Although containing rich information, most of the data is too complicated to be understood by human brains, and therefore considerable interest has been placed onto extracting maximum value from the available data resources and infrastructures.

Data mining, or machine learning, is about creating computer programs that automatically mine through datasets and identify their hidden patterns. The unearthed true patterns should generalise well to make predictions for the future. In the real world, the collected data is almost always dirty/noisy due to various errors in the data generation and collection processes, and therefore data mining algorithms need to be sufficiently robust to distinguish regularity from accidental coincidences [1].

People who apply data mining techniques often have the following two major goals: a) accurate forecasting of what will happen in the future by learning from the data describing the past events, b) the patterns extracted by data mining methods, which are used for prediction, are also compact descriptions of the underlying structure of the data. Those patterns, which are often represented as linear equations, rules or tree architectures, can offer users with useful insights into the particular problems [2].

As an applied discipline, data mining has established as popular analytic techniques in a wide range of application domains, including fraud detection [3], credit scoring [4], cancer diagnosis and prognosis [5, 6], drug sensitivity prediction [7], etc.

This doctoral thesis aims to develop novel algorithms for various data mining problems using mathematical programming-based optimisation techniques.

1.1 Data Classification and Regression

In data classification, a dataset is given in the form of a n x p data matrix, where measurements are recorded for n samples/instances (rows) and p features/attributes (columns). Depending on the particular problems in hand, samples may be medical patients, companies or chemical reactions, and features can be clinical measurements of the patients, financial performance of companies or environmental variables of reactions. Each sample is also annotated to one of a pre-defined set of classes, e.g. patients who survive a surgery and patients who die from the surgery, bankrupt companies and non-bankrupt companies and reactions of high productivity, medium productivity and low productivity.

Given a dataset, a classification method, or classifier, learns a mapping f from p features to class label. Classification is a supervised learning process because the categories of the samples are known. For most of the classifiers in literature, the mapping f, representing the underlying functional form of the classification method, has a pre-specified structure with some unknown parameters. Training a classifier refers to estimating those parameters of f so as to minimise a suitable cost function on the given samples. Cost function can be defined as the number of misclassified training samples, i.e. number of samples that the estimated mapping f fails to classify correctly. The estimated functional mapping f can then be used to determine the class label of new samples, providing only their measurement values on p features. The above procedure of solving a classification problem is shown in Figure 1.1 below.

Classification problems can be broadly divided into two main categories: binary and multi-class classification problems. The number of pre-defined classes

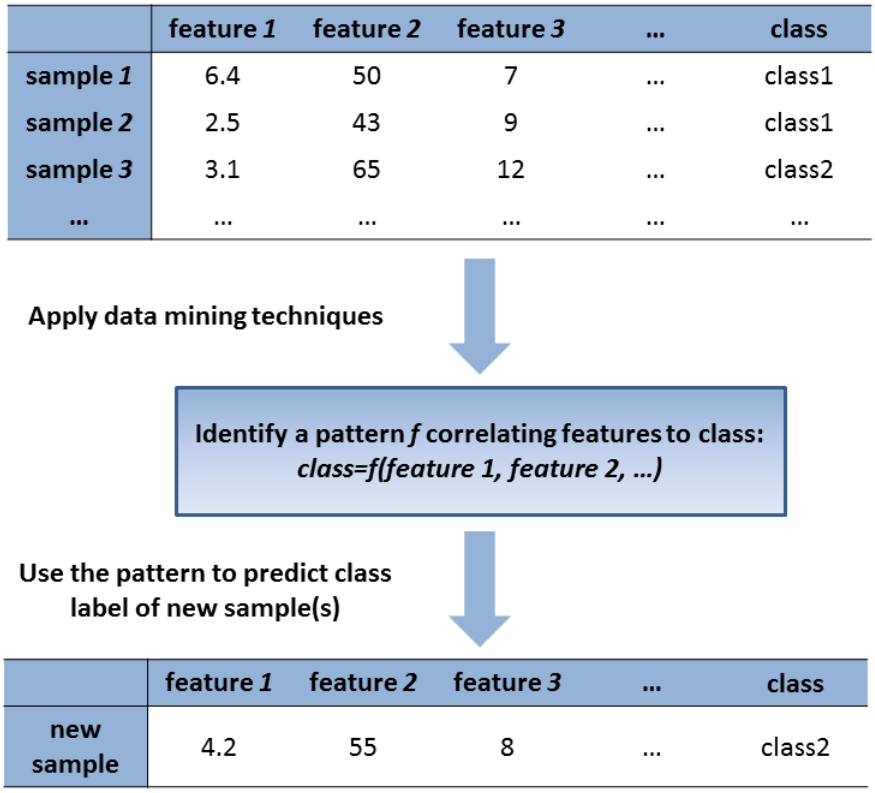

FIGURE 1.1: Overview of procedures of solving a classification problem.

is two for binary classification problems and more than two for multi-class problems. Many of the traditional classification methods were originally devised for binary classification problems [8], including the well-known support vector machine (SVM) [9] and linear discriminative analysis (LDA) [10]. Common strategies of tackling a multi-class classification problem include either solving the problem once using a multi-class classification algorithm or decomposing the problem into a series of binary problems and solving them using binary classifiers [11, 12].

While classification predicts categorical outcomes, regression analysis forecasts output variables that take numerical values. Cost functions in regression analysis can be any arbitrary monotonically increasing function of deviation, i.e. the difference between predicted outcome value from the model and real outcome value. In practice, minimisation of squared or absolute errors are most frequently employed to derive parameters of the regression functions.

A large number of learning algorithms have been proposed for classification and/or regression analysis, such as multi-layer perception (MLP) [13–15], SVM, K-nearest

neighbours (K-NN) [16, 17], multivariate adaptive regression splines (MARS) [18] and tree classification and regression models [19, 20]. Furthermore, those standalone learning algorithms can be integrated into an ensemble model, who produces a single prediction by weighting and combining predictions made by the single algorithms. Ensemble models are shown to often achieve more robust performances than single methods [21].

In this thesis, we propose a number of novel standalone machine learning methods for both classification and regression tasks, the performances of which are benchmarked against a number of state-of-the-art methods in open literature.

1.2 Disease Classification using High-Throughput Genomic Data

Despite continuous research effort, the development and progression of complex diseases remain poorly understood. Take breast cancer as an example, although certain clinical variables have been demonstrated to correlate with prognosis to some extent, including lymph node metastases, histological grade of tumour, tumour size and mutation in the *TP53* gene, they do not yield accurate personalised prognosis for patients [22]. Furthermore, patients with the same disease status give disappointingly varying responses to the same treatments and develop different overall outcomes [23, 24]. Given the difficulty of accurate personalised prognosis and treatment, the current clinical practice observes that the majority of early stage breast cancer patients receive adjuvant therapies, including chemotherapy and/or endocrine therapy [25]. Many of them, however, would have survived without the aggressive treatments. It is estimated that during the past three decades about 1.3 million women in the United States alone were over-treated, causing significant long-term side effect on patient and healthcare cost [26]. It is clear that accurate personalised prognosis and treatment prediction are needed to assist clinicians on offering the optimal therapeutic strategies [27].

Microarray profiling technology enables simultaneous examination of expression levels of thousands of genes for each patient in a single chip, and a cohort study typically contains multiple patients [28]. The millions of data points generated per

cohort study carry rich information to study complex diseases, for example cancer and psoriasis, from gene level [29]. In one of the landmark works [30], breast cancer patients are clustered into four distinct groups, based on the similarity in their gene expression patterns. This study has suggested that breast cancer is a heterogeneous collection of diseases with distinct molecular aberrations, changing the traditional view that breast cancer is a single gene disease [23].

Classification techniques have also been widely applied to identify the dependence between gene expression and the clinical outcomes of interest using the gene expression data, with genes being the feature, patients being the sample and clinical outcomes of patients being the class label. Two separate prognostic gene signatures [6, 31], i.e. sets of genes, have been developed, with genes being selected on the basis of differential expression between two phenotypic outcomes, i.e. distant recurrence group and non-distant recurrence group. The two gene signatures outperform the traditional pathological variables in predicting 5-year distant metastasis for individual patients. The gene signature constructed in [31] has been commercialised as MammaPrint and made available to patients in the United States since 2007. Another gene signature [27] has been proposed that stratifies breast cancer patients into three distinct clusters, including those who survive without adjuvant treatments, those who require and respond well to the treatment and those die despite the treatment. Similar work exists in literature that proposes diagnostic or prognostic signatures for other complex diseases of unknown mechanisms, including acute leukemias [32], diffuse large B-cell lymphoma (DLBCL) [33], psoriasis [34] and lung cancer [35].

The major difficulty associated with classification with gene expression data lies upon the inherent "large p small n" nature of the high-throughput genomic data, whereby the number of samples is usually two orders of magnitudes smaller than the number of genes in a single transcriptomic profile, making it hard to extract reliable information [28]. A close examination of the two landmark gene signatures [6, 31] find that the prognostic accuracy of gene signature derived from one patient cohort is significantly lower when validating on the other cohort [36]. Furthermore, despite similar numbers of genes, the two signatures only share three common genes. This lack of consensus and reproducibility between gene signatures derived from different patient cohorts has later been confirmed as a common phenomenon [37]. On the other hand, a recent study [38] has shown that more than

half of the published gene signatures fail to provide statistically better prediction than random gene signatures. Those observations indicate that the current gene signatures cannot lead to biologically interpretable insights into the underlying mechanism of breast cancer progression, albeit the predictive value. It is argued that at least several thousand samples are needed to achieve a desired level of robustness when deriving gene signatures from microarray profiles [37].

In order to identify biomarkers that offer higher predictive accuracy and more biological insights, many recent publications have integrated biological knowledge as *a priori*, typically in the forms of either biochemical pathways or protein interaction network (PIN), into the analysis of microarray genomic data, and therefore approaching the problem of disease classification from the level of gene set. Each biochemical pathway is an expert curated collection of genes that interact with each other to perform specific cellular processes, for example metabolism, membrane transport, signal transduction and cell cycle [39]. A human protein interaction network is a graphical representation of interactions of genes or gene products, where nodes represent genes or gene products and edges denote functional interactions between them. Tackling disease classification problems on gene set level provides several benefits: 1) the difficulty of analysing high dimensional transcriptomic profiles is much alleviated since the genes fall within common pathways or connected to each other on PIN are remarkably smaller than the number of genes characterised in microarray [40–42], 2) combining different sources of expert knowledge can greatly enhance the reliability of the discoveries [43], and 3) disease genes do not act alone, rather collectively modulate cell fate, thus evaluating functional gene sets is more likely to yield biologically interpretable biomarkers that help with understanding of disease mechanism [44, 45].

To date, gene set level analysis has already been demonstrated to produce encouraging results [40]. For example, methods have been proposed that successfully identify some well-known oncogenic pathways, and suggest certain pathways or gene modules as potentially disease relevant [46–48]. On the other hand, gene set signatures are shown to outperform or at least match conventional gene signatures in terms of both prediction accuracy and reproducibility across different patient cohorts [41, 42, 48, 49]. Those results have clearly suggested that gene set level analysis, by incorporating external biological knowledge, represents a step forward in terms of finding robust biomarkers and releasing the true underlying mechanism

of disease progression.

In this thesis, we present a gene set level computational approach for disease classification problems, by incorporating pathway information. A novel optimisation-based computational method is proposed that leads to improved diagnostic and prognostic accuracies than the existing methods in literature.

1.3 Clustering of Directed Networks

Networks are fundamental representations of many complex systems. At the basic level, a network is made up of a set of nodes and edges connecting pairs of nodes. Edges in the networks can be weighted, directed and signed, with the weights, directionality and sign respectively indicating the strength, direction and activation or inhabitation of the relation. The rich amount of information that a network representation can carry makes it a popular tool to model World Wide Web [50], social network [51, 52], metabolic network [53, 54] and so on. In World Wide Web network, html pages act as nodes and hyperlinks denote directed edges. In metabolic networks, nodes can be various cellular constituents while edges are chemical reactions that sustain the cellular functions. Over the past two decades, extensive research has released several interesting topological properties that commonly exist in large-scale real world networks, for example degrees of nodes, i.e. the number of direct neighbours, generally follow power law distribution [55] and two arbitrary nodes in a network are usually connected via a relatively small number of intermediate nodes [56].

A particularly useful topological property of many real world networks is the presence of strong community structure, whereby the nodes can be clustered into communities, or modules, with much denser within-community interactions than cross-community interactions (Figure 1.2) [57]. Each community can be viewed as a discrete building block of specific functionality, and discovery of those communities provides insights into the formation of the network and the underlying principles of the dynamics of the system [57, 58]. Studying the latent community structure has already yielded many valuable discoveries. For example, in Web network, pages that share similar topics are more likely to belong to the same

community and understanding the inherent community structure of the Web network helps build better search engines. Analysing the communities in metabolic networks may identify the key components that are densely connected to other components and maintain the integrity of the network.

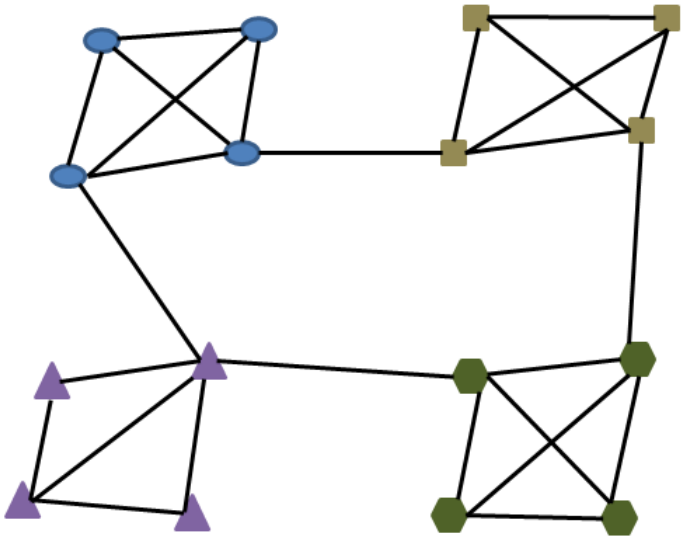

FIGURE 1.2: Network community detection.

1.3.1 Community detection in undirected networks

Newman and Girvan [59] propose a metric, called modularity, which measures the quality of network division for undirected networks. Informally, modularity (Q) is defined as the fraction of the within-community edges in a network minus expected fraction of the within-community edges for the same network with random placements of edges. High positive values of modularity suggest that, for the given network division, the quantity of within-community edges are statistically higher than what would be expected by chance. On the other hand, low modularity values suggest that the network division is close to random. Since the proposal of this modularity function, a number of community detection algorithms have been presented that seek the optimal division of nodes into communities so as to maximise the modularity metric. Due to the combinatorial nature of the problem, searching of the optimal network division is known to be NP-complete, and therefore almost all practical algorithms in literature are based on certain heuristics, greedy search,

simulated annealing and so on [58, 60–64].

1.3.2 Community detection in directed networks

While most research on community structure detection is focusing on the simple case of undirected networks, relatively little has been done for networks where edges are directed. In reality, directionality of the edges often carry indispensable information about the underlying complex systems. For example, in a citation network where nodes are academic journal publications and a directed edge denotes existence of citation relationship, discarding the direction of the citation relationships and treating the network as undirected is improper [65]; in a gene regulatory network where genes are nodes and an edge represents the regulation from one gene to the other, clearly ignoring the useful information on directionality will likely lead to inaccurate findings [66].

Leicht and Newman [57] adopt the modularity function for undirected network and generalise it to directed network by explicitly considering both the in-degree and out-degree distributions of nodes, where in-degree and out-degree respectively denote the numbers of edges pointing to the node and pointing from the node. A few algorithms that were originally devised for modularity optimisation of undirected networks have been modified for identifying community structure in directed networks [57, 60, 67, 68].

In this work, a mathematical programming model is introduced for optimising modularity function in division of directed networks. Observing that the amount of computational resource required to achieve a quality solution of the models grows exponentially with the size of the problem, an efficient heuristics-based solution procedure has also been proposed in this thesis. Applied to benchmark directed networks from different application domains, the proposed algorithms have been demonstrated to outperform the state-of-the-art methods by identifying network divisions of noticeably higher modularity.

1.4 Mathematical Programming Optimisation Techniques

Mathematical programming, or mathematical optimisation, is applied throughout this thesis to tackle the above problems. In general, mathematical programming is an optimisation technique that formulates a given problem as a mathematical model and identifies the optimal solution corresponding to the maximised/minimised objective function value [69].

A typical mathematical programming model looks as below:

$$\begin{aligned} min \quad & f(x) \\ s.t. \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in X \end{aligned}$$

where $x \in X \subseteq R^q$ are the decision variables, $f(x)$ is the objective function, and $g(x) \in R^r$ and $h(x) \in R^s$ are the inequality and equality constraints, respectively. Given a specific problem, one needs to identify the decision variables and formulate the objective function and constraints. The constraints and subset X define a feasible region, where there are multiple feasible solutions. The purpose of mathematical programming optimisation is to find, within the entire feasible region, the best set of values for decision variables so that the objective function is maximised/minimised.

Mathematical programming models are classified into the following categories based on the type of variables:

- Linear programming (LP): a LP model consists of linear objective function and constraints, where all decision variables are continuous;
- Non-linear programming (NLP): there is non-linearity in objective function and/or constraints, where all decision variables are continuous;

- Mixed integer linear programming (MILP): similar to LP, while some of the decision variables are restricted to take discrete values, i.e. 0, 1, 2, ...;
- Mixed integer non-linear programming (MINLP): similar to NLP, while some of the decision variables are restricted to take discrete values.

Once a mathematical programming model is formulated and programmed in a computer, certain general purpose solution algorithms can be employed to solve them and identify the solutions. A lot of those algorithms exist, including the revised simplex algorithm for linear programming formulations, interior point algorithm for non-linear programming, and branch & bound and cutting plane algorithms for mixed integer programming models. Thanks to the rapid computational development, a large number of commercial software are currently available to program mathematical models in computer recognisable formats, such as GAMS [70], AIMMS [71] and EXCEL, and solve them by calling solvers that contain the above algorithms, for example CPLEX, GUROBI, BARON, etc. Unless stated otherwise, all the implementations in this thesis are conducted in GAMS 24.0 and executed on a 3.20 GHz and 12.0 GB RAM desktop computer.

Solving large-scale mathematical optimisation problems are known to be computationally expensive. LP models are theoretically demonstrated to be solvable in only weakly polynomial time [72], while NLP, MILP and MINLP are generally non-convex and therefore even harder to solve [73]. Thus, in most of the literature and also this thesis, efficient solution procedures are proposed to achieve quality locally optimal solutions at reasonable computational cost.

1.5 Scope and Contribution of the Thesis

In the midst of rapid accumulation of various sources of data, how to build algorithms that automatically and efficiently extract the maximal value of those data by identifying the implicit patterns has recently emerged as a popular research avenue. We address several important topics in this thesis, including multi-class data classification, disease classification using high-throughput genomic data, regression analysis and modularity maximisation in directed network, by proposing novel mathematical programming-based optimisation models and efficient solution procedures. The major contributions of this thesis are outlined as below:

1.5.1 Multi-class Data Classification

In classification problems, one seeks to accurately classify samples into one of the pre-defined set of categories by inferring the relationship between features and category. In this thesis, the problem of multi-class classification is studied, where the number of pre-defined categories is more than two. An existing mathematical programming-based multi-class classifier [74] in literature is adopted and two novel solution procedures are proposed to achieve solutions of better quality. To demonstrate the applicability and efficiency of the proposed classifier, a number of benchmark classification problems are employed, and the performance of this novel classifier is studied against other state-of-the-art approaches.

1.5.2 Disease Classification using High-throughput Genomic Data

The arrival of cost-effective and reliable high-throughput gene expression data has made it possible to propose computational frameworks for disease diagnosis and prognosis. Observing the fact that incorporating trustable biological knowledge as *a prior* has generally proven to be beneficial in literature, this thesis also tackles pathway level disease classification. Briefly, a novel mathematical optimisation model is presented that summarises, for each pathway, the expression patterns of its constituent genes into a composite feature, before performing classification using the resulting composite features. Several datasets covering complex and multifactorial diseases, e.g. breast cancer, prostate cancer, psoriasis, have been used to test the robustness of our novel approach against other pathway-based and traditional genes-based classification methods.

1.5.3 Data Regression

Similar to data classification, regression analysis aims to estimate the relationships between several independent input variables/features and output variables, which take continuous values. In this thesis, a novel piece-wise linear regression method is firstly presented that partitions one feature into multiple mutually exclusive regions and fit for each region one distinct multivariate linear regression function.

The location of break-points and regression coefficients of each region are simultaneously optimised by a novel mathematical programming model.

The piece-wise linear regression method is subsequently generalised to a tree-based regression model. Instead of partitioning one single feature into multiple intervals, this tree-based regression model performs recursive binary partitions, using the same optimisation model as proposed and restricting the number of non-overlapping regions to two. The binary partitions are recursively performed on different features and thus ensures that it can better accommodate complex local non-linearity. Benchmark datasets have been used to demonstrate that the two novel regression methods match or outperform the popular regression methods in literature in terms of predictive accuracy.

1.5.4 Network Division in Directed Network using Modularity Maximisation

On modularity optimisation of directed network, an MINLP formulation is firstly presented. Exact linearisation of the non-linear constraints is performed, which results in an MILP model. The MINLP formulation can quickly realise locally optimal solutions while the MILP model theoretically guarantees global optimal solution but is practically solvable for only small networks. Therefore, a two-step hybrid system is introduced that first solves the MINLP model to yield a quality initial partition of the network, before iteratively removing the module memberships of some nodes and re-allocating them by solving the MILP models. The iterative procedure continues until the modularity value converges. This hybrid system has been shown to convincingly outperform the existing network clustering methods in literature.

1.6 Thesis Structure

The rest of this thesis is structured as below:

In Chapter 2, multi-class data classification problem is addressed. An existing classifier based on hyper-box principle is introduced, before two refined solution

procedures are described. Comparative studies are conducted to comprehensively evaluate the performance of the new classifiers.

Chapter 3 focuses on a particular type of classification problem, i.e. data classification on high-dimensional datasets. Assigning the features into different pathways according to prior biological knowledge, a novel optimisation model is presented that linearly combines the features in a pathway into a new composite feature, whose discriminative power is maximised. A large number of real disease datasets have been used to demonstrate the desirable performance of the introduced classification framework.

Chapter 4 and 5 address the regression problem. After introducing a base mathematical programming model partitioning a single feature into multiple regions and fitting one linear function per region, two separate solution procedures are employed that produce a piece-wise linear regression method and a tree-based regression method. Again benchmark examples widely tested in literature are used to calibrate the prediction performance of those two approaches against other popular counterparts.

Chapter 6 describes a new hybrid optimisation framework for clustering nodes in a directed network into different non-overlapping communities so that to maximise modularity of the partition. In the two-step procedure, the initial step consists of solving a number of times a computationally cheap MINLP model and collecting a quality solution for the second step. In the second step, some nodes are released and allocated to other modules by solving a linearised version of the MINLP model. The second step is conducted in an iterative manner until modularity does not improve.

Finally, Chapter 7 concludes with the major findings of this thesis and outlines the future directions.

Chapter 2

Mathematical Programming-based Classification Models

Data classification refers to the problem of applying computational models that, when presented the input feature values and output class labels for various samples, automatically identify hidden relationships which can be used to assign unseen samples into one of these classes. Mixed integer programming optimisation techniques have been frequently used in literature to design classification models. In this chapter, an integer programming-based classifier [74], modelling decision boundary as hyper boxes, has been adopted from literature and two novel solution procedures have been proposed that lead to improved performance of the methods.

2.1 Introduction and Literature Review

Given a set of samples, each of which is described by certain measurable features and labelled with a pre-determined class, data classification concerns identifying the pattern within the data and predicting the class labels of new samples. Data classification has a wide range of applications from financial analysis [75–77], image classification [78–80], medical data for disease diagnosis or prognosis [81–83], market price prediction [84] and document classification [85, 86].

Over the past decades, a wide range of classification algorithms have been proposed in literature to tackle various classification problems. Classification algorithms can be broadly divided into two categories: binary and multi-class classifiers. A binary classifier is solely applicable to classification problems with two classes while a multi-class classifier can deal with problems with more than 2 classes. Compared with the large number of binary classifiers, there are relatively fewer multi-class classifiers in literature [87]. Common strategies of tackling a multi-class classification problem include either solving the problem once using a multi-class classification algorithm or decomposing the whole problem into a series of binary problems and solving iteratively the sub-problems using binary classifiers [88].

The existing classifiers in open literature are based on diverse methodologies, including support vector machine (SVM), neural network (NN), naïve Bayesian, decision tree, mathematical programming optimisation techniques, and so on. We provide below a brief summary of some of the most popular classification approaches, with some key classifiers shown in Figure 2.1.

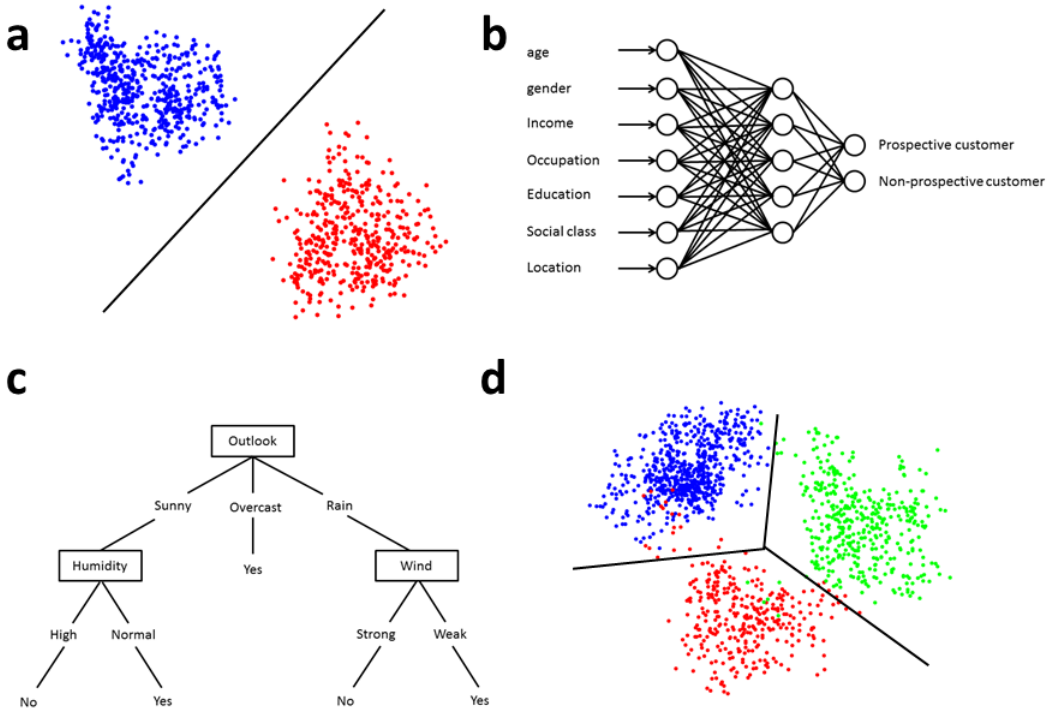

FIGURE 2.1: Some key classifiers in literature. a: SVM; b: NN; c: decision tree; d: piece-wise linear classifier

2.1.1 Support Vector Machine (SVM)

SVM constructs hyper planes to separate samples from different classes. SVM builds hyper planes under the condition of maximum soft margin, i.e. maximising the margin between two classes while allowing certain amount of misclassifications of the samples. The balance between distance of the constructed hyper plane to different classes of samples and the amount of misclassifications is controlled by a user-specified trade-off parameter. One of the features that make SVM powerful is the so-called kernel trick, which maps the dataset to higher-dimensional inner product space, where samples may be easier to separate. A number of kernel functions, which greatly enhance the suitability of SVM in modelling non-linear decision boundaries, can be employed, e.g., polynomial kernels and radial basis function kernel. Solving SVM has been formulated as a convex quadratic programming optimisation problem, which can be solved to global optimality.

SVM has been originally designed as a binary classification algorithm, and the approach of decomposing a multi-class problem into a series of "one vs. one" or "one vs. all" sub-problems are commonly employed for it to be used for multi-class classification tasks [89]. Despite the popularity, optimal tuning of the trade-off parameters and choice of kernel functions remain problem-specific issues that considerably affect the prediction power of SVM [90, 91].

2.1.2 Neural Network (NN)

Mimicking a biological neural network, NN classifier consists of a number of connected layers of neurons, which transforms an input layer of features to an output layer of class labels. Each neuron takes input as weighted summation of outputs from all the neurons in the previous layer, and applies a non-linear activation function before passing the output to all the neurons in the next layer [92]. Frequently used activation functions include: sigmoid, logarithmic and radial basis functions [93]. Despite its capacity to tackle datasets with non-linear and complex decision boundaries, the number of hidden layers, how many neurons allowed for each hidden layer, which activation function to use amount to a difficult optimisation problem, which limits the generality of the method [94]. In reality, the structure of the network, i.e. the number of layers, the number of neurons for each layer and the types of activation function, are usually specified by the user, which reduces

the problem of training a neural network classifier to tune the weights of connections between consecutive layers of neurons to minimise the classification error. Training a neural network is known to be time consuming and can only guarantee local optimality.

2.1.3 Naive Bayes

Naïve Bayes classifier belongs to the group of statistical classifiers. It is based on the simple assumption that the effect of different features on class membership predictions is independent from each other [95]. In general, Naïve Bayes simply computes the support of each feature for each class so that the maximum likelihood estimate is satisfied in the training samples set. With the derived Bayesian rules the probability of a sample being predicted into a class can be calculated. The simplicity of Naïve Bayes classifiers also ensures computational efficiency [96]. Although the assumption of independence among features is more often than not violated in practical datasets, naïve Bayesian generally gives comparable performance against much more sophisticated classifiers [97].

2.1.4 Decision Tree

Decision tree is a recursive partitioning method that sequentially splits samples into subsets. Starting from the whole dataset, decision tree identifies one attribute and a break point, before partitioning samples into subsets so as to improve the homogeneity of the class label vector within the subsets. The partitioning procedure is recurred for each child node until no further split can result in an increase in training sample accuracy [98]. After growing a large tree, small leaves that do not contribute significantly to the training accuracy are removed to improve the generalisation of the constructed tree [99]. Interpretability is one of the main strengths of decision tree classifiers. The set of sequential linear rules generated are easy to understand, providing valuable insights into the mechanism of the underlying system. Decision tree has been shown to be particularly vulnerable that perturbing a small proportion of training samples or re-sampling the training set are likely to result in a very different tree structure.

2.1.5 Mathematical Programming-based Classifiers

Another group of classification models are built on mathematical programming optimisation techniques. Gehrlein [100] proposes a classification formulation in which each class is given a linear function, i.e. a linear combination of attributes. For a given sample, one score is computed for each class by fitting its attribute values into its linear function. A sample is considered correctly classified when the score of its true class is greater than the scores of the other classes. The model optimises the coefficients of linear functions so as to maximise the number of correctly classified samples. With the same concept of assigning one linear function per class. Recently, Bal & Orkcu [87] employ the technique of goal programming in their formulation. Given a sample, when the score of its true class is less than the score of another class, misclassification happens and there is a positive deviation equalling to the difference between the two scores. The objective function is to minimise the total deviations of misclassified samples, instead of the number of misclassifications. The model is formulated as a linear programming problem, and therefore having the advantage of requiring little computational cost. Sueyoshi published similar works on linear discriminative functions for binary or multi-class classification problems, including [75, 77].

Ryoo [101] proposes a model that simultaneously constructs a number of hyper planes forming piece-wise linear decision boundary for binary classification problems. The two classes of samples are respectively in either side of the decision boundary. The number of hyper planes is a user-defined tuning parameter. Furthermore, the choice of which class of samples to be enclosed in the convex region of the piece-wise linear boundary also requires manual intervention. Lastly, the proposed formulation is solely applicable to binary classification tasks. Bagirov et al. [102] also focus on building piece-wise linear classifier to separate samples. Their proposed method firstly singles out a set of problematic instances that are hard to classify correctly, followed by building piece-wise linear planes with the number of the planes being incrementally increased. Bertsimas & Shioda [103] present a model separating samples into a number of polyhedrons, which are formed by multiple hyper planes. The class of a polyhedron is the one having the greatest number of samples inside the polyhedron. The proposed formulation tries to enclose as many samples belonging to the same class into the same polyhedrons by optimising the positions of polyhedrons.

On the other hand, Xu & Papageorgiou [74] produce a mathematical programming-based formulation modelling a hyper box (HB) classifier. A hyper box is essentially a multi-dimensional rectangle with the number of dimensions being equal to the total number of attributes in the dataset. The proposed method aims to build for each class a number of hyper boxes enclosing as many samples as possible. The hyper boxes belonging to different classes are constrained to not overlap with each other, and each hyper box defines a distinct rule enclosing a proportion of training samples. In [104], a modified version of HB classifier has been developed which requires only 1/3 to 1/2 computational time compared with the original HB. Inspired by the promising performances of the HB classifier, we propose a refined hyper box classifier in this work, aiming to improve the quality of the constructed boxes.

2.1.6 Ensemble Classifiers

Besides the single classifiers described above, some recent research efforts have been focusing on developing ensemble classifiers, which train a number of classifiers and aggregate their classification outcomes to produce the final prediction. Given a training sample set, Bagging [105] creates a number of bootstrap sample sets by uniformly sampling with replacement, and each bootstrap sample set is then learned by a classifier. The final prediction is an aggregation of decisions made by each classifier, via either simple average or more sophisticated voting strategy where certain classifiers have more votes in the final decision [106]. Another recent advance in ensemble classification algorithm is Boosting. One of the most recognised Boosting algorithms is Adaboost [107], which trains a set of classifiers in an iterative manner so that the subsequent classifiers are constructed in favour of those samples misclassified by the last classifier, by updating the weight distribution of samples. Given a new sample with unknown class label, all the single classifiers make their own predictions of which class it belongs to and their decisions are combined to yield a final prediction.

In the following sections, both mathematical formulation and solution procedure of the original HB classifier are firstly summarised, before two novel solution procedures are introduced in a bid to improve the performance of the original classifier.

Comparative studies have been done using benchmark datasets to demonstrate the efficiency of the new classifiers.

2.2 A Hyper Box Classifier in Literature

As mentioned, this work is based on the classification method proposed in [74], which describes a mathematical programming formulation modelling hyper box classifier. A hyper box is a multi-dimensional rectangle and belongs to one class. Multiple hyper boxes can be potentially created for each class. A sample is called correctly classified when enclosed in (at least) one hyper box of its class. Training of the hyper box classifier involves determining the coordinates of all the hyper boxes so as to maximise the number of samples that can be enclosed in their corresponding hyper boxes, subject to the constraint that hyper boxes belonging to different classes are prohibited to overlap, i.e. occupy the decision space. Then the coordinates of the hyper boxes represent decision boundary between different classes.

The original formulation is reviewed in the below subsection before the proposed refinements are presented.

2.2.1 Original Mathematical Formulation of HB

The indices, parameters and variables associated with the model are listed below:

Indices	
s	sample
m	feature/attribute
i,j	hyper box
i_s	hyper box i that sample s is mapped into
c,k	class/category
i_c	hyper box i that belongs to class c
Parameters	
A_{sm}	numeric value of sample s on feature m
C	the total number of classes
M	the total number of features
U	an arbitrarily large positive number
ϵ	an arbitrarily small positive number
Free variables	
X_{im}	central coordinate of hyper box i on feature m
Positive variables	
LE_{im}	length of hyper box i on feature m
Binary variables	
E_s	*1* if sample s is correctly enclosed in its hyper box; *0* otherwise
Y_{ijm}	*1* if on feature m lower bound of hyper box i is greater than upper bound of hyper box j and thus ensuring non-overlapping between the two boxes; *0* otherwise

Whether a sample s is enclosed in its corresponding hyper box i_s or not is modelled using the following two sets of constraints:

$$A_{sm} \geq X_{im} - LE_{im}/2 - U(1 - E_s) \qquad \forall s, i_s, m \tag{2.1}$$

$$A_{sm} \leq X_{im} + LE_{im}/2 + U(1 - E_s) \qquad \forall s, i_s, m \tag{2.2}$$

If E_s takes the value of *1*, sample s is correctly enclosed in hyper box i_s, i.e. value of A_{sm} lies between the lower bound ($X_{im} - LE_{im}/2$) and upper bound ($X_{im} + LE_{im}/2$) of its hyper box i_s for *all* attributes; otherwise sample s is misclassified as being outside its target box. In Figure 2.2 a, two dimensional representation of samples being inside and outside their corresponding hyper boxes are provided.

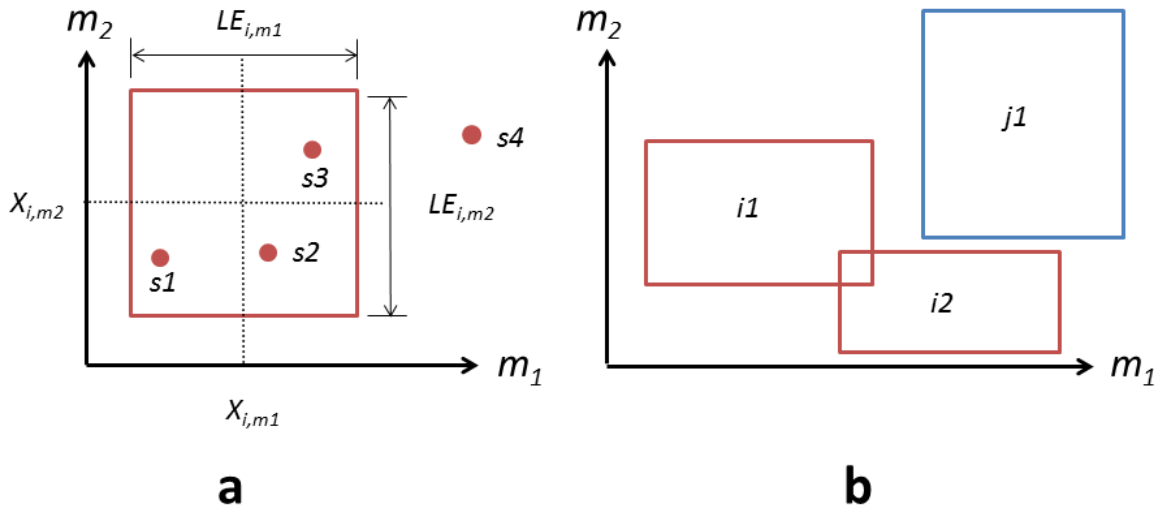

FIGURE 2.2: Graphic explanations of mathematical formulation of HB. a: sample *s1*, *s2* and *s3* are correctly enclosed in its hyper box (i.e. $E_s = 1$) while sample *s4* lies outside the box (i.e. $E_s = 0$); b: Non-overlapping constraints are enforced for hyper boxes belonging to the same class.

Hyper boxes of different classes are not allowed to overlap, which is realised via the following two sets of constraints:

$$X_{im} - X_{jm} + UY_{ijm} \geq (LE_{im} + LE_{jm})/2 + \epsilon \qquad \forall m, i_c, j_k, c \neq k \tag{2.3}$$

$$\sum_{m} (Y_{ijm} + Y_{jim}) \leq 2M - 1 \qquad \forall i_c, j_k, c < k \tag{2.4}$$

In Equ. (2.3), when binary variable $Y_{ijm} = 0$, hyper box i and j belonging to different classes are constrained not to overlap, because lower bound of box i is greater than upper bound of box j on attribute m; when binary variable $Y_{ijm} = 1$ Equ. (2.3) become redundant. To avoid overlapping of the hyper boxes in M-dimensional space, they need to not overlap in at least one dimension, which is modelled by Equ. (2.4). In Figure 2.2 b, a graphical example of overlapping and non-overlapping hyper boxes is given. The objective function is to minimise the number of misclassifications (i.e. $E_s = 0$):

$$min \quad \sum_{s} (1 - E_s) \tag{2.5}$$

The final formulation, named MCP in the original paper [74], is made up of objective function Equ. (2.5) subject to sample enclosing constraints Equ. (2.1) and (2.2), and hyper box non-overlapping constraints Equ. (2.3) and (2.4). The combination of linear objective function and constraints, and the presence of binary variables define an MILP formulation, which can be solved to global optimality using standard solution techniques, for example branch-and-bound.

2.2.2 Iterative Solution Procedure of Hyper Box Classifier

The last section describes a mathematical programming formulation for building hyper boxes to separate samples. In [74], an iterative solution procedure has also been developed to allow potentially multiple hyper boxes per class to improve the quality of the solution. The original iterative procedure is outlined in Figure 2.3 below.

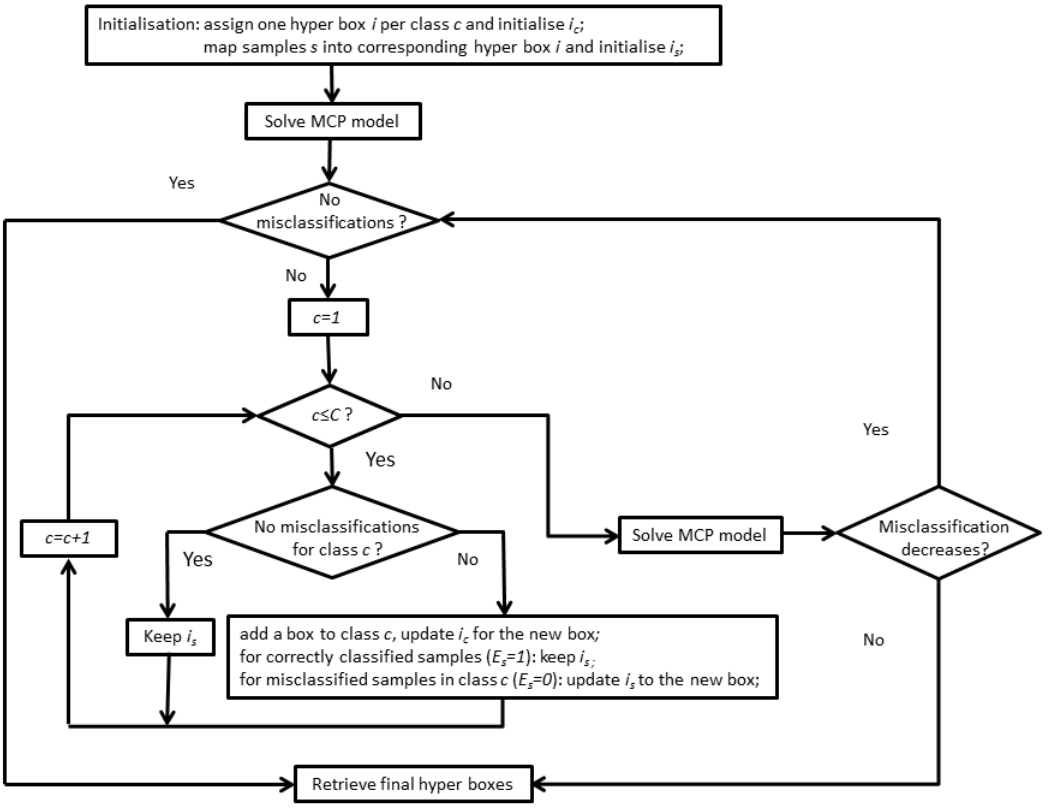

FIGURE 2.3: HB iterative solution procedure

Initially, one hyper box is created for each class of samples (initialise i_s) and the MCP model is solved once to enclose as many as possible the samples into their

own hyper boxes. Starting from the second iteration, for any class having at least one misclassified sample ($E_s = 0$), one additional hyper box is allowed for this particular class, followed by updating i_s, i.e. the correct classified samples are still mapped to their original hyper box while the misclassified samples are re-mapped to the new box. For the classes that all their samples are correctly classified in the last iteration, their sample-box mapping are kept. The iterative procedure terminates when the number of misclassified samples does not decrease in two adjacent iterations or when all the samples are correctly classified. An artificial illustrative example is given in Figure 2.4 to illustrate the old iterative solution procedure.

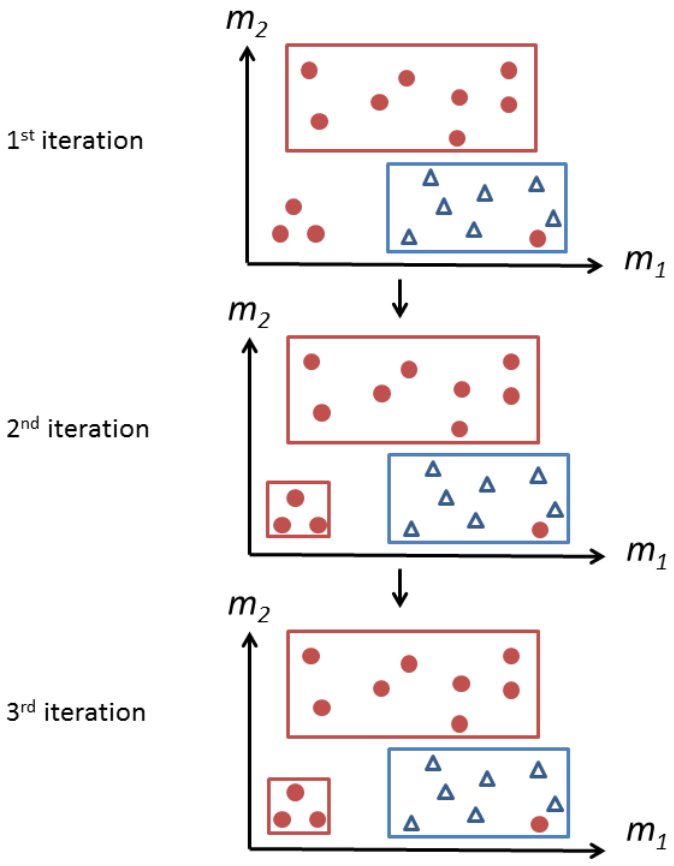

FIGURE 2.4: HB iterative solution procedure.

2.2.3 Predicting New Samples using Derived Hyper Boxes

After training the HB classifier the derived hyper boxes are used to predict the class label of a new sample. The prediction procedure is intuitive as: 1) if a new sample falls into one of the derived boxes, it is assigned the class label of the box;

2) if a new samples lies outside all derived hyper boxes, it is assigned the class label of its nearest box, based on Euclidean distance.

After reviewing the main features of hyper box classifier proposed in [74], a refined HB classifier will be proposed in the next section.

2.3 A Sample Re-weighting Hyper Box Classifier

Inspired by the idea of boosting algorithms, which typically consists of iteratively learning classifiers while updating the weight distribution of samples, a sample re-weighting scheme is introduced into the traditional hyper box classifier in a effort to improve its performance.

As mentioned earlier in the previous section, the traditional HB inherently involves iterative training, i.e., after each iteration any class with misclassified samples is updated with an extra hyper box and the MCP model is re-solved. The proposed method mimics the behaviour of boosting algorithms by re-weighting samples between iterations. More specifically, after each iteration, we update the weights of all samples by assigning higher weights to a subset of misclassified samples, thus putting more emphasis into correctly classifying them in the next iteration. When a sample s is misclassified by its hyper box, the misclassification can fall into two categories: 1) misclassified sample lies outside all derived boxes; 2) misclassified sample lies inside at least one of the derived boxes that belong to a different class. In this study, the two types of errors are termed type 1 and 2, respectively. Figure 2.5 visualises the two types of misclassifications for a two dimensional case.

In Figure 2.5 a, two misclassified samples lie outside both derived hyper boxes and before the next iteration, another box will be allocated for the two samples of type 1 error. In the second iteration, the two samples will be correctly enclosed in the additional hyper box. In Figure 2.5 b, however, the two type 2 misclassified samples will still be misclassified in the next iteration despite another allocated hyper box. In fact, type 2 misclassified samples have only slight chance of being correctly classified in the following iterations. In this work, a sample re-weighting scheme that gives more weights to the type 2 misclassifications is proposed, which will increase the chance of them being correctly classified and achieving a better final solution. In order to accommodate the different weights of samples, the

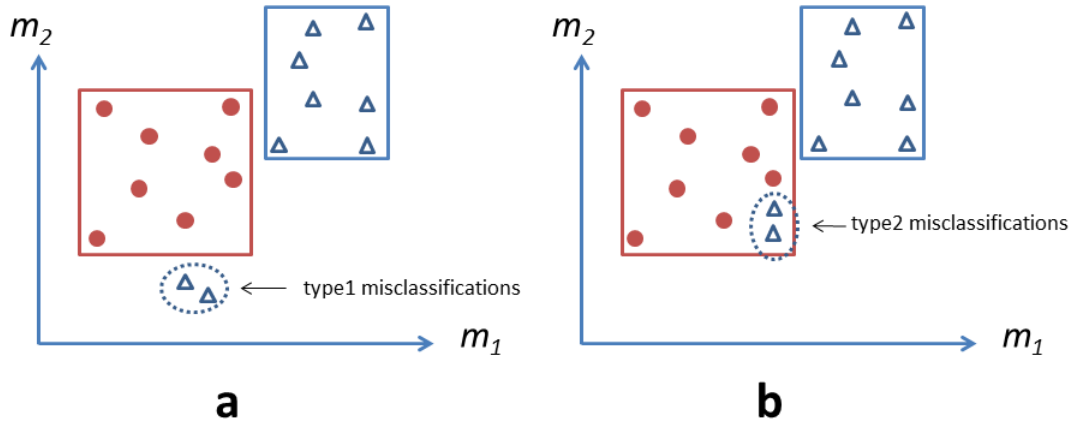

FIGURE 2.5: Two types of hyper box misclassifications. a: type 1 misclassification that samples are not enclosed correctly by its hyper box and are outside all the boxes from other classes; b: type 2 misclassification that samples are not enclosed correctly by its hyper box and are inside at least one of the boxes belonging to another class.

objective function Equ. (2.5) in the traditional HB has been modified to the following:

$$min \quad \sum_{s} P_s(1 - E_s) \tag{2.6}$$

where P_s denote the weight of sample s, equivalent to the cost of misclassification. Equ. (2.5) can be seen as a special case of Equ. (2.6) where $P_s = 1$ for all samples. Considering the new objective function, when different weights are assigned to different samples, the model will prioritise those samples with higher weights for the overall misclassification cost to reach globally minimum. We keep other constraints Equ. (2.1)-(2.4) in the new formulation, which is named W_MCP (Weighting_MCP). The W_MCP formulation is still an MILP. The flowchart of the modified iterative hyper box method, called SRW_HB (Sample Re-Weighting_Hyper Box), is constructed in Figure 2.6:

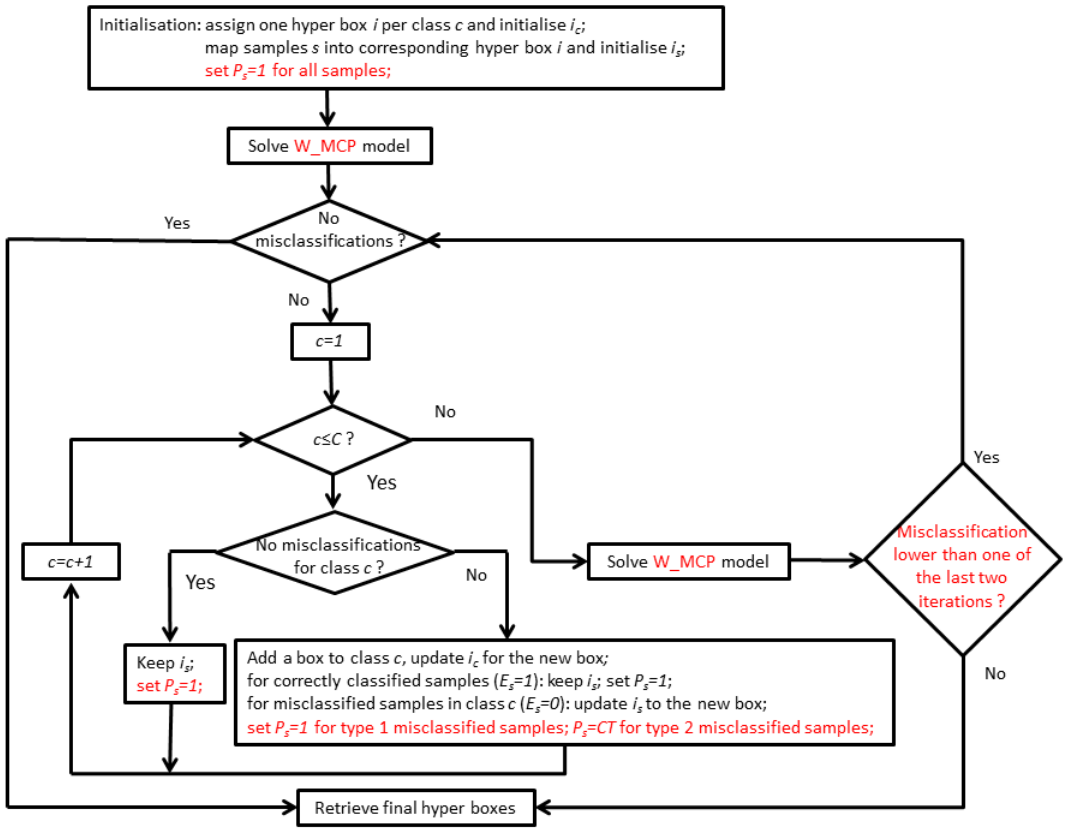

FIGURE 2.6: Flowchart of the proposed SRW_HB. The highlighted content in red differentiates the SRW_HB from the traditional HB.

The proposed SRW_HB also implements an iterative solution procedure. The first iteration is identical to the first iteration of the traditional HB that one box per class is generated to minimise the total cost of misclassifications while all the samples are having a weight value, P_s, of *1*. If there are misclassified samples, from the second iteration one more box is allowed for each class with at least one misclassified sample. The sample-box mapping is updated that correctly classified samples from the last iteration keep their mapping from the last iteration, while the misclassified samples (both type 1 and 2) are re-mapped to their newly generated hyper boxes. The misclassification cost for correctly classified samples and type 1 misclassified samples are set to *1*, while the cost for type 2 misclassified samples are set to a higher value CT ($CT > 1$). The W_MCP model is re-solved and the above procedure is repeated. The iterative solution procedure terminates when the number of misclassified samples fail to improve in 2 consecutive iterations. The testing procedure is the same as the original HB that a new sample is allocated to its nearest derived hyper box and then assigned the membership of the hyper box.

2.4 A Data Space Partition Scheme

In the original publication [74], it is claimed that for some datasets, MCP models cannot be solved to global optimality in 200s for all iterations. Note that computational complexity of an MILP problem is dependent on the size of the problem,a data space partition scheme is proposed to ease the computational burden of building hyper boxes and attempt to identify better solutions.

Given a dataset A_{sm}, the average value of all samples on each attribute m is calculated as $Aver_m$, followed by computing the number of samples satisfying $A_{sm} \geq Aver_m$ and $A_{sm} < Aver_m$, respectively, which are denoted as RU_m and RL_m. Compute for each attribute the difference between the numbers of samples placed in the two disjoint regions partitioned from $Aver_m$ as $Diff_m = |RU_m - RL_m|$. The attribute offering the most even partition, i.e. the smallest $Diff_m$ value is selected as the partition attribute $m*$. When there are multiple attributes offering equally low $Diff_m$ value, the partition attribute is randomly chosen among them. Subsequently the original dataset is partitioned into two disjoint regions *R1* and *R2*, which respectively contain samples satisfying $A_{sm*} \geq Aver_{m*}$ and $A_{sm*} < Aver_{m*}$. In each region, we train the proposed sample re-weighting hyper box classifier (SRW_HB). It is important to note that extra constraints are added to the W_MCP to make sure that the derived hyper boxes from each region are not unnecessarily large to overlap with hyper boxes derived from the other region on the partition attribute $m*$, thus ensuring the boxes in one region do not overlap with the boxes in the other region:

$$X_{im} - LE_{im}/2 \geq Aver_m \quad \forall i, m = m* \tag{2.7}$$

$$X_{im} + LE_{im}/2 \leq Aver_m - \epsilon \quad \forall i, m = m* \tag{2.8}$$

Equ. (2.7) are added to W_MCP when solving $R1$ while Equ. (2.8) are added to W_MCP when training on samples in $R2$. An arbitrarily small positive constant ϵ is inserted in Equ. (2.8) to ensure the two regions do not share the same boundary. The final decision boundary is formed by all the derived hyper boxes from both regions. The idea behind the data space partition method is that the required computational time to solve an MILP grows exponentially with the number of training samples, making it hard to identify optimal solutions at feasible

computational cost. Partition the dataset into two disjoint regions with similar numbers of samples makes both regions roughly equally easy to solve. We name the framework employing the proposed simple data space partition scheme to create two disjoint sub-regions and construct sample re-weighting hyper box classifiers in both regions as DR_SRW_HB, the flowchart of which is illustrated in Figure 2.7.

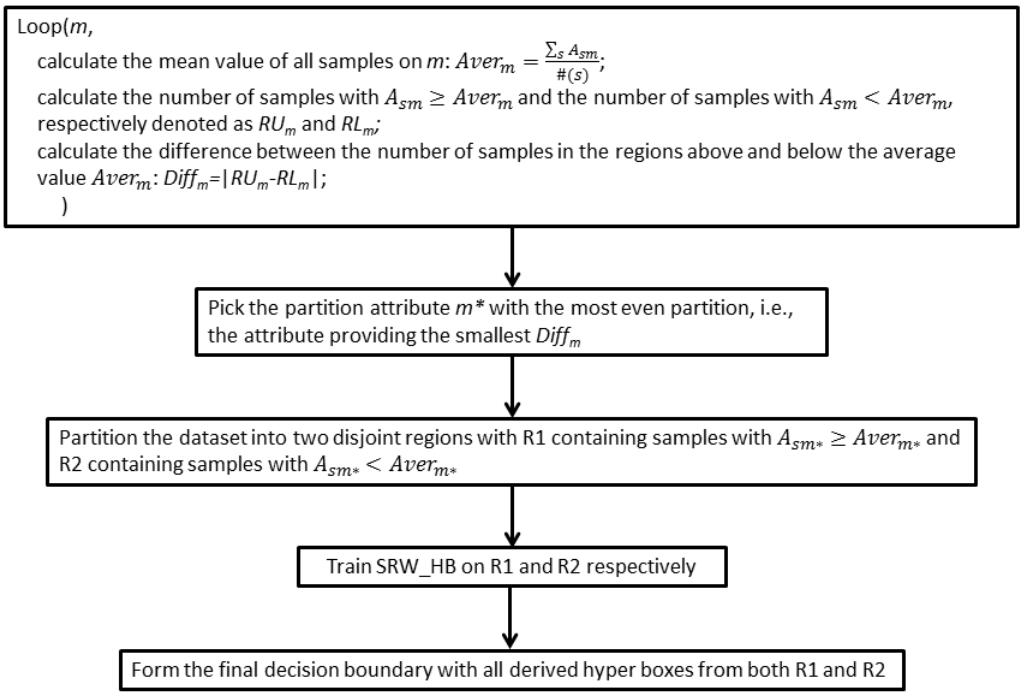

FIGURE 2.7: Flowchart of the proposed DR_SRW_HB

In this work we have tested the proposed data space partition scheme, which splits the entire data space into two disjoint regions, on medium-size datasets. It is important to note that for larger size datasets, the current proposed strategy can be further generalised, i.e., partition the data space into 3,4 or more disjoint parts, to accommodate more samples and attributes.

2.5 Computational Results

In this section, the applicability and effectiveness of the proposed SRW_HB and DR_SRW_HB classifiers are demonstrated through 6 real world datasets, including Phenol [108], Firm [74] and 4 datasets downloaded from UCI machine learning repository (http://archive.ics.uci.edu/ml/), namely Ionosphere, glass, breast tissue, and iris. We have implemented a number of literature classifiers to compare the classification rates with our proposed SRW_HB and DR_SRW_HB. The group of classifiers include Naïve Bayes, Sequential minimal optimisation (SMO, which is a realisation of support vector machine), Logistic regression, Bagging, Adaboost, NN and three mathematical programming-based multi-class classifiers: HB [74], Bal & Orkcu [87] and Gehrlein [100]. To comprehensively evaluate the overall classification performances of various classification algorithms, we use two testing scenarios as below:

Scenario 1: perform 50 random partitions of each dataset into a training set containing 70% samples and a testing set containing the 30% samples. For each partition we train a classifier on training set and test the classification performance on testing set.

Scenario 2: conduct a leave-one-out cross validation that for each dataset hold only one sample in the testing set while using the rest as training samples. The process is repeated until all samples are used as testing sample.

All the mathematical programming-based classification methods, including SRW_HB, HB, and approaches proposed by Bal & Orkcu [87] and Gehrlein [100], are implemented in General Algebraic Modeling System (GAMS) 24.1 [70] and solved using CPLEX 12.3 solver on a 2.40 GHz speed, 2393 MHz cpu computer system. Optimality gap is set as *0* when solving MILP problems. For all hyper box-based methods we limit the computational time per iteration as 200 cpu seconds.

Other classifiers are implemented in Waikato Environment for Knowledge Analysis (WEKA) machine learning software [109]. Default setting are retained for Naïve Bayes, Logistic regression, SMO, Bagging and Adaboost. For SMO, default setting include Complexity parameter C=1, polynomial kernel of exponent of 1

is used. Bagging is done on REPtree with 10 iterations, with bag size percentage being equal to 100 %. For adaboost, DecisionStump is used as base classifier with 10 iterations. With regards to NN, the following parameters from [74] are used: hiddenLayers=2; learning rate=0.1; momentum=0.7; trainingTime=10000. In WEKA, default setting for NN specifies one single hidden layer with the number of neurons equalling to the sum of the numbers of features and classes divided by 2.

2.5.1 Real World Datasets

We use 6 real world datasets to test the applicability and competitiveness of the proposed classification algorithms. Ionosphere concerns some radar data that given 34 attributes reflecting the received signals the task is to classify free electrons in the ionosphere into 2 classes. The dataset Phenol [108] concerns classifying 274 phenols, characterised by 9 molecular descriptors that quantify their compounds, into 4 possible toxicity mechanisms including polar narcotics, respiratory uncouplers, pro-electrophiles and soft electrophiles. Glass example is a collection of glass samples belonging to 6 types of glass. Each sample is described by 9 attributes, each of which corresponds to weight percentage of a chemical compound (sodium, aluminium, calcium etc.) in corresponding oxide. Breast tissue dataset has 106 freshly excised tissue samples in the breast area, and are descripted by 9 attributes such as area under spectrum, length of the spectral curve. Iris is one of the most studied benchmark datasets in data classification. 150 instances from 3 types of iris plant are characterised by 4 features, including sepal length, sepal width, petal length and petal width. Firm dataset aims to predict the financial performance of a number of companies, based on certain performance indices for example cash to total assets, long-term debt to total assets, into a class of 'good' firms and the other class of firms went bankrupt between 1996 and 2002. A brief summary of the employed real world datasets is provided in Table 2.1.

2.5.2 Sensitivity Analysis of CT

In this section, a sensitivity analysis is performed to tune the user-specific parameter CT for the proposed SRW_HB, which denotes the cost for type 2 misclassified

TABLE 2.1: Summary of real world datasets

dataset	Number of samples	Number of attributes	Number of classes
Ionosphere	351	34	2
Phenol	274	9	4
Glass	214	9	6
Breast tissue	106	9	6
Iris	150	4	3
Firm	83	13	2

samples and is higher than *1*. We present in Figure 2.8 the results of sensitivity analysis for all the 6 datasets.

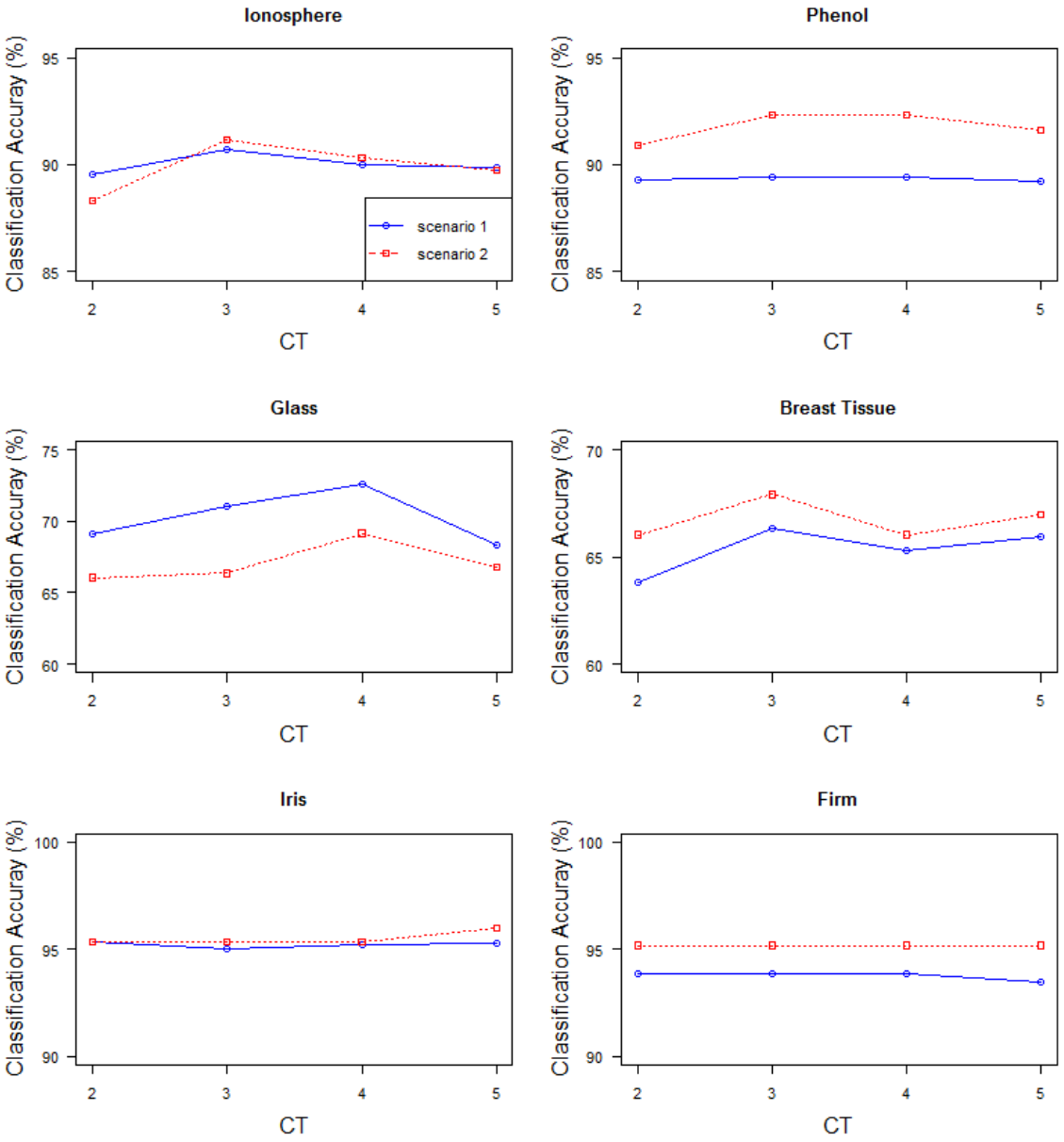

FIGURE 2.8: Sensitivity analysis of CT for the proposed SRW_HB on two testing scenarios. Blue line with circle markers denotes scenario 1 and red line with rectangle markers denotes scenario 2.

A series of values have been tested for CT, including *2, 3, 4* and *5*. It is clear from Figure 2.8 that varying CT has different effects on different datasets. For Ionosphere datatset and scenario 1, prediction accuracy first increases from $CT = 2$ to $CT = 3$, and then falls down when CT is equal to *4* and *5*. With regards to scenario 2, the trend is similar that prediction rate goes up from $CT = 2$ to $CT = 3$, and then decreases later on. For Phenol, as CT increases classification rate for scenario 2 goes up from $CT = 2$ to $CT = 3, 4$ before decreasing when $CT = 5$,

while classification rates for scenario 1 keep constant. Glass is the most affected by different values of CT among all tested datasets that for both scenarios prediction rates increase from $CT = 2$ to *4* by about 4%, which subsequently drops down when $CT = 5$. With regards to Breast tissue case study, classification rates for both scenarios fluctuate throughout the tested CT values and both peaked at $CT = 3$. When it comes to Iris dataset, increasing CT appears to have minor impact on scenario 1, while for scenario 2 prediction rate keeps constant between $CT = 2$ and *4* before growing slightly with $CT = 5$. Lastly, for Firm dataset, classification rate for scenario 2 keeps constant over the tested range while for scenario 1 the accuracy goes down from $CT = 4$ to *5*.

Overall, it is obvious that the sensitivity analysis for SRW_HB does not yield a clear optimal CT value, as in different datasets and different scenarios peak prediction rates come from different CT values. On the other hand, it appears that $CT = 3$ gives a robust performance as prediction rate often peaks at or near $CT = 3$ (e.g. Ionosphere, Breast Tissue). Therefore we take $CT = 3$ for SRW_HB when comparing its classification performance against other implemented classifiers in literature, which has good performance for almost all datasets investigated.

2.5.3 Classification Accuracy Comparison

In this section, we evaluate the classification performance of 10 classifiers, including the proposed SRW_HB and traditional HB. For the proposed SRW_HB classifier, we set $CT = 3$ for all datasets to offer a fair comparison. The results are presented in Table 2.2 and 2.3 for scenario 1 and 2, respectively.

Table 2.2: Classification rate comparison for scenario 1

Classifier/dataset	Ionosphere	Phenol	Glass	Breast tissue	Iris	Firm
SRW_HB	90.69%	89.41%	**71.09**%	66.32%	95.64%	93.84%
HB	89.37%	87.02%	68.53%	63.16%	94.76%	**93.92**%
Gehrlein [100]	84.55%	86.05%	56.68%	52.32%	93.64%	86.67%
Bal & Orkcu[87]	89.15%	84.80%	61.59%	59.23%	86.93%	89.20%
Naïve Bayesian	82.52%	86.29%	48.13%	**67.34**%	96.00%	92.61%
SMO	87.94%	81.29%	55.84%	54.31%	**96.67**%	93.16%
Logistic regression	86.48%	88.07%	62.16%	64.56%	95.56%	86.73%
Bagging	**90.96**%	**90.17**%	68.69%	66.75%	94.67%	92.43%
Adaboost	90.36%	77.85%	42.78%	36.19%	94.36%	93.16%
NN	88.77%	87.99%	59.97%	60.66%	95.11%	93.23%

*The highest accuracy for each dataset is highlighted in bold, same for Table 2.3

Table 2.3: Classification rate comparison for scenario 2

Classifier/dataset	Ionosphere	Phenol	Glass	Breast tissue	Iris	Firm
SRW_HB	91.17%	**92.34**%	66.36%	67.92%	96.00%	95.18%
HB	89.74%	90.51%	65.89%	66.98%	94.00%	95.18%
Gehrlein [100]	84.55%	86.05%	56.68%	52.32%	93.64%	86.67%
Bal & Orkcu [87]	87.18%	87.59%	64.95%	**68.87**%	88.67%	**98.80**%
Naïve Bayesian	82.62%	86.50%	49.53%	66.04%	95.33%	91.57%
SMO	88.03%	79.56%	54.67%	56.60%	96.67%	95.18%
Logistic regression	89.17%	89.05%	62.62%	**68.87**%	**98.00**%	84.34%
Bagging	**92.02**%	91.24%	**72.90**%	65.09%	94.00%	90.36%
Adaboost	90.88%	78.47%	44.86%	40.57%	97.33%	93.98%
NN	89.46%	88.69%	59.35%	60.38%	95.33%	91.57%

For both testing scenarios, no classifiers are showing dominant classification rates against others, as different datasets play to the strengths of different classification methodologies. This observation is consistent with the previous findings [110, 111]. A good classifier should maintain consistently good performance across many different classification problems. The proposed SRW_HB, showing this desired consistency, is usually among the top 3 out of the 10 classifiers. Note that the proposed SRW_HB outperforms the traditional HB for most scenarios.

We summarise here the overall classification performance of the 10 implemented classifiers by using a scoring scheme, employed also in [74]. Briefly, for each scenario and a particular dataset, the classifiers are ranked in descending order according to their prediction accuracies, i.e. the classifier with the highest classification rate is awarded a score of 10; the classifier with the second highest classification rate is assigned a score of 9, and so on. For each scenario, the average score across all datasets is taken as the indication of the overall competitiveness of a particular classifier. The higher the average score, the better the performance of the classifier.

The score ranking is presented in Figure 2.9, which shows that in both scenarios the proposed SRW_HB classifier not only gives improved classification accuracy from the traditional HB, but also outperforms other state-of-the-art classifiers.

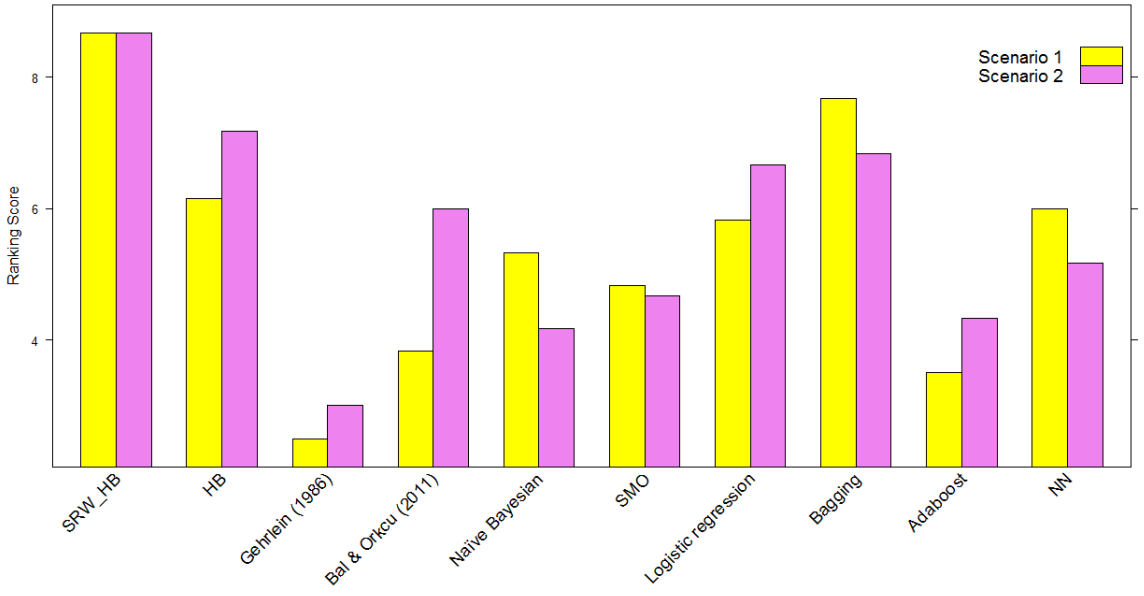

FIGURE 2.9: Overall standing of classifiers. In both scenarios, the proposed SRW_HB leads to the most robust classification performance across all implemented classifiers.

2.5.4 DR_SRW_HB significantly reduces computational cost while maintaining the classification accuracy compared with SRW_HB

In the last section, it is demonstrate that the proposed SRW_HB classifier, which modifies the traditional HB classifier by updating the misclassification costs for

samples with type 2 errors after each iteration, gives overall better prediction accuracy compared with a number of state-of-the-art classifiers. Recall that we have proposed in Section 2.4, a DR_SRW_HB method that implements a simple data space partition scheme to split the original data space into two disjoint regions, followed by training the SRW_HB for both regions. The idea is that each region contains about half samples of the entire problem, which is then much easier to solve.

We now test the effectiveness of the data space partition scheme, by comparing the performance of DR_SRW_HB against SRW_HB for both scenarios. With regard to the proposed SRW_HB method, the underlying model cannot be solved to global optimality for at least one iteration (within 200 s) on 4 datasets (either scenario), including Phenol, Glass, Breast tissue and Ionosphere. We therefore run the space partition-based DR_SRW_HB on those 4 datasets and compare the prediction accuracies with these achieved by SRW_HB. The results are presented in Table 2.4. For scenario 1, DR_SRW_HB leads to higher classification rate on Phenol while SRW_HB is more accurate on Glass, Breast tissue and Ionosphere. It should be noted that compared other literature classifiers, DR_SRW_HB still shows better overall performance. When it comes to scenario 2, DR_SRW_HB offers much higher prediction rates on Glass example, ties with SRW_HB on Phenol and Breast Tissue while losing on Ionosphere example. We can see that DR_SRW_HB performs better in scenario 2 than scenario 1, because scenario 2 requires more computational effort than scenario 1 as a result of more samples involved in training of scenario 2,. Considering both two scenarios, it is therefore conclusive that the proposed data space partition scheme can maintain the overall prediction rates of SRW_HB on complex examples.

Table 2.4: Classification rate comparison between two proposed classifiers DR_SRW_HB and SRW_HB

Scenario 1	Ionosphere	Phenol	Glass	Breast tissue
DR_SRW_HB	90.08%	90.41%	69.19%	63.55%
SRW_HB	90.69%	89.41%	71.09%	66.32%
Scenario 2	Ionosphere	Phenol	Glass	Breast tissue
DR_SRW_HB	89.74%	92.34%	73.36%	67.92%
SRW_HB	91.17%	92.34%	66.36%	67.92%

Recall that the DR_SRW_HB has been proposed to overcome the high computational cost of tackling complex classification problems, we report here, for scenario 2, the average computational time per run consumed by three variants of hyper box-based classifiers, namely HB, SRW_HB and DR_SRW_HB. The results, presented in Figure 2.10, clearly show that by partitioning a complex problem into two sub-problems and solving two relatively easy problems, the computational cost dramatically decreases. On Phenol and Breast tissue, DR_SRW_HB constructs hyper boxes in a matter of seconds while the CPU time consumed by HB and SRW_HB are significantly higher. While it takes hundreds of seconds for DR_SRW_HB to train hyper boxes on Glass and Ionosphere, the actual computational time is still small fractions of the consumption of HB and SRW_HB. For scenario 1, the trend is similar that the proposed data space partition method considerably reduces computational cost (data not shown).

We also compare our proposed DR_SRW_HB with an alternative solution procedure proposed in literature for hyper box classifier [104], in which after each iteration, correctly classified training samples are removed and the dimensions of established hyper boxes are fixed before optimising the hyper boxes for the next iteration. It has been shown that the proposed solution procedure results in the computational cost saving of 2 3 fold and generally decreased classification accuracy. Our proposed DR_SRW_HB classifier clearly outperforms [104] by offering much higher computational cost reduction. Thus, it is concluded that DR_SRW_HB results in huge CPU savings of 1 or 2 orders of magnitude, compared with HB and SRW_HB. Overall, we propose here a strategy that for a classification problem which SRW_HB struggles to identify globally optimal solutions for all iterations, the DR_SRW_HB is used instead; otherwise for an easy classification problem, the SRW_HB is used.

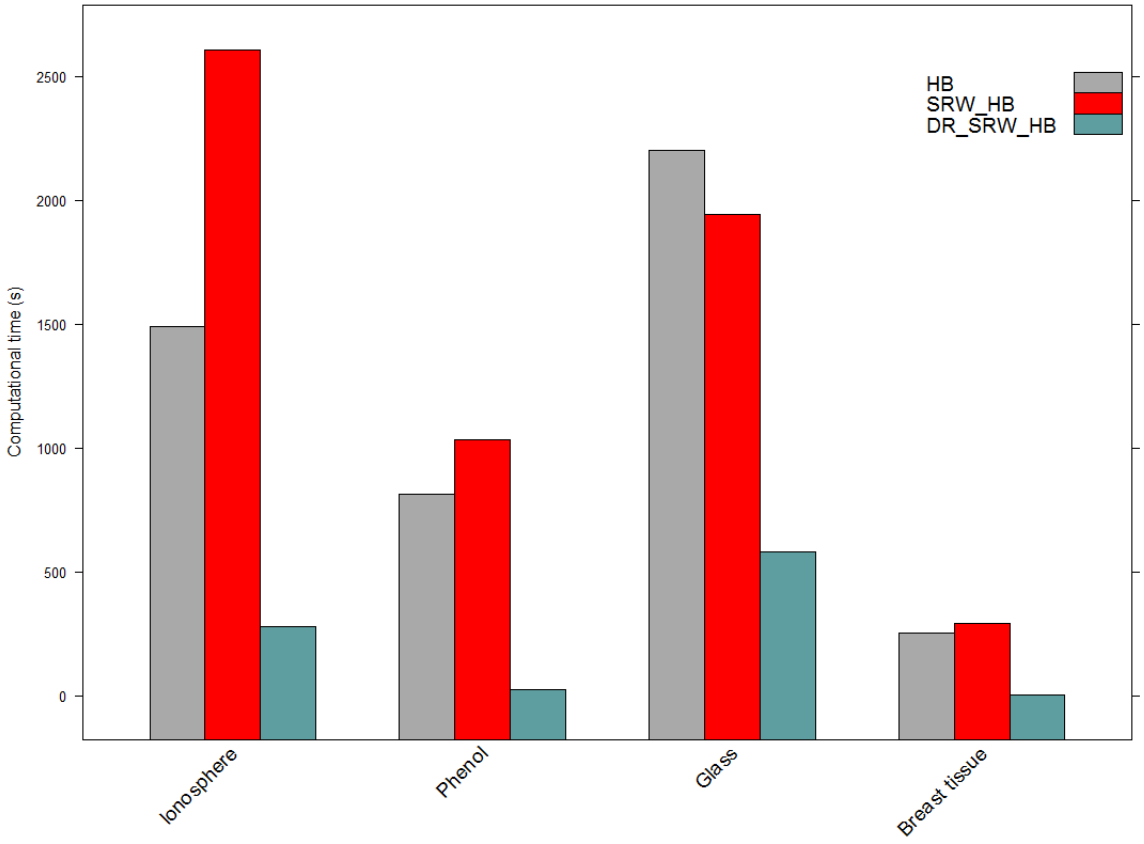

FIGURE 2.10: Computational cost comparison between HB, SRW_HB and DR_SRW_HB. In the figure, average computational time per run of scenario 2 is reported for traditional HB, SRW_HB and DR_SRW_HB on 4 datasets Phenol, Glass, Breast tissue and Ionosphere.

Despite the significant reduction in computational time, DR_SRW_HB, based on mixed integer programming, is still generally consuming more computational resource than the existing methods in literature. We note here that the prediction accuracy remains the most important aspect of many real world data classification problems, for example medical disease classification problems [81]. The classifiers proposed in this work are aimed to achieve higher prediction accuracy for off-line classification problems where computational time is not of major concern.

2.6 Concluding Remarks

Data classification is an important data mining area subject to extensive on-going research interest. Inspired by the promising classification rates of a hyper box classifier [74] in literature, we propose in this work two new solution procedures that aim to improve the performance of hyper box classifier. The first improvement, SRW_HB, updates the samples weights during each iteration of the training process so that the type 2 misclassified samples, i.e. misclassified samples enclosed in one of the hyper boxes from another class, are given more weights than the other samples. Through 6 binary and multi-class real world datasets, it is demonstrated that the proposed SRW_HB can provide consistently good classification rates, outperforming the traditional HB and other state-of-the-art classifiers for example SVM, NN and Logistic regression.

A data space partition method has also been introduced to reduce the computational cost of SRW_HB, which works by splitting the dataset into two disjoint regions, each of which is then solved independently using SRW_HB. On the 4 complex datasets, the proposed DR_SRW_HB appears to consume dramatically less computational time than the original HB and SRW_HB, often in 1 to 2 orders of magnitude, on the basis of maintaining the desirable level of prediction accuracy compared with the proposed SRW_HB classifier.

A natural extension of this work in the near future is to investigate a more generic data space partition scheme. The sample partition scheme presented and used for DR_SRW_HB proves to significantly reduce computational cost but can only perform binary partition. For large-scale data classification problems, the proposed DR_SRW_HB may struggle to identify quality solution in training procedure. Therefore, a generic data space partition method, which splits data into multiple regions and each one of which is easy to solve, can help scale up the hyper box classifiers to large-size problems.

Furthermore, classification accuracy, defined as the percentage of correctly predicted samples, is used as the metric for comparing the performance of various competing methods in this work. It is noted here that when it comes to a practical problem, more comprehensive examination of the performance of a classifier should be conducted on the basis of individual samples to yield better insights.

For example, identify the set of consistently misclassified samples may help release the weakness of a classifier, i.e. the range where the particular classifier does not predict well.

Chapter 3

Pathway-level Classification of Complex Diseases using High Throughput Gene Expression Profiles

For complex diseases, traditional clinical variables are shown to poorly correlate with patients' phenotypic outcomes. High throughput microarray profiling technology analyses, in a single snapshot, the expression levels of thousands of genes for all patients in the cohort. This chapter focuses on developing a novel computational framework that accurately classify patients into their corresponding phenotypic outcomes using gene expression data.

3.1 Introduction and Literature Review

For complex diseases, the current diagnostic and prognostic factors fail to accurately distinguish patients of different clinical outcomes [6, 24, 27, 30], mainly because of the heterogeneous nature of disease aetiology and poorly understood underlying mechanisms [112]. The advent of high throughput methods, where snapshots of gene or gene product activity (feature) are profiled in samples of varying disease states (class), has provided a powerful means of relating disease phenotypes to latent genetic mechanisms. In charactering disease, the gene expression matrix serves as input to a classification task where each sample is allocated

to a relevant phenotypic class via specific gene signatures or biomarkers that can best differentiate between outcomes. Such disease classification tasks have been successful in deriving biomarkers for diagnosis [113], prognosis [6, 25, 114] and response to treatment [115, 116] in complex disorders.

Despite successful reports, disease classification is impeded by the so-called "large p small n" property, whereby the number of samples is typically several orders of magnitude smaller than the number of genes (features), making it difficult to extract reliable information from the transcriptomic profiles. Therefore, to obtain statistically reliable gene signatures capable of accurate prediction of samples into their corresponding phenotypic outcomes, suitable reduction of dimensionality is sought. Dimension reduction aims to reduce the number of features in the original gene expression matrix to a size that is conducive for efficient disease classification. Traditional computational methods estimate the discriminative power of individual genes, whose expression pattern can best distinguish the samples, and a classifier is subsequently applied on the reduced set of differentially expressed genes. However, gene-based disease classification approaches have been shown in various recent studies to yield non-reproducible gene signatures of moderate prediction accuracy [117, 118]. Thus, current studies have increasingly focused on integrating biological data a priori, including protein interaction networks (PIN) [42] or biological pathway information [41] into gene expression data and proposing biomarkers at the level of functional sets of genes. It has been demonstrated that biomarkers represented as groups of genes result in higher prediction accuracy, more reproducibility across different datasets and better mechanistic interpretation [42, 48, 119].

In the below subsections, both traditional gene-based disease classification approaches and more recent gene set-based approaches are briefly reviewed.

3.1.1 Single Gene-based Approaches

In general, disease classification approaches where all genes are considered simultaneously employ a feature selection algorithm to select a subset of highly differentially expressed genes from the entire set of profiled genes, before training a classifier on the selected genes. In [6] and [31], genes were rank-ordered by their degree of correlation with the breast cancer distant-metastasis-free survival

time and breast cancer outcome respectively, and gene signatures have been constructed by sequentially adding genes from the ranked lists until the maximum prediction performance is reached. A similar sequential gene selection method has been described in [120], which takes into account both the discriminative power of individual genes and their correlation. The underlying idea is that addition of a new candidate gene into the optimal gene set must contribute towards increased classification performance while maintaining a low level of correlation with the current genes in the set to reduce redundant information. An optimisation model that outputs a user-specified subset of genes is proposed in [121], where group of genes is selected to achieve maximal cross-class separability, minimal same-class tightness and gene pair-wise correlation. Similar methods are proposed in other studies [33, 81, 82, 113, 122].

A number of ensemble methods are also available, which combine the advantages of many different classification methods, so as to derive a more efficient framework in comparison to stand-alone feature reduction and classifiers. For example, principal component analysis is employed in [123] to project the expressions of genes into 10 dominant principal components, followed by inner cross validation procedure where an artificial neural network classifier was trained for each training sample subset. All constructed classifiers subsequently cast a vote to determine the phenotype of a testing sample. In [124], a list of genes with best discriminative power is constructed and then assigned to different gene subsets. A neural network classifier is trained with each subset of genes and the final ensemble classifier is formed with majority voting strategy. Given a training sample set, the method from [125] creates a number of bootstrap sample sets by drawing with replacement and determining for each bootstrap sample set the weights of genes. Ensemble feature ranking is achieved by aggregating gene weights over all bootstrap sample sets to produce a final gene ranking. In [126], information gain is used to evaluate both the separability of genes and gene-gene dependence, followed by clustering genes into different gene groups with a Markov blanket. Subsequently, different gene sets are constructed by randomly sampling one representative gene from each gene group, and an ensemble classifier is built by learning from each gene set. Similar ensemble-based approaches using microarray samples for disease classification are reported in [127–129].

Gene markers constructed across different datasets share disappointingly little

overlap, despite similar predictive power [42, 117, 118]. This lack of agreement limits the applicability of such methods and renders mechanistic interpretability problematic. Such diagnostic or prognostic profiles relate to genes that do not act in isolation, but in fact work in concert, forming sub-networks that collectively modulate or determine cell fate. Accounting for such molecular synergies in feature reduction and disease classification protocols can also alleviate challenges of single-gene classifiers related to cellular heterogeneity in tissue, genetic heterogeneity among patients, measurement noise, thereby leading to increased biological interpretability of biomarkers and enhancing insights into the mechanisms of the disease. Therefore, feature selection and classification methods where all genes are treated independently are increasingly replaced by approaches where the effects of groups of genes on disease prediction are considered simultaneously. Such gene sets can either reflect curated biochemical pathways or functional modules derived from protein interaction networks, discussed in the following sections.

3.1.2 Network-based Approaches

The past few years have seen remarkable growth of protein interaction data. Network principles, where nodes represent genes or their products and edges indicate some type of interaction between them, have served as a particularly suitable abstraction basis to develop computational procedures for understanding system properties [130]. Disease module-based methods assume that all cellular components that belong to the same topological, functional or disease module have a high likelihood of being involved in the same disease [131]. This strategy involves constructing the interactome by integrating available data from online databases in the tissue or cell lines of interest and then identifying functional units that contain most of the disease-associated genes. Disease modules are then validated by, for example, showing that the genes in a module have related functions or correlated expression patterns.

Several methods have been proposed that extract functional modules of genes, whose expression patterns can distinguish samples from different phenotypes [42, 119, 132]. Analysis of two breast cancer patient cohorts have revealed that altered modularity of the human interactome may be useful as an indicator of breast cancer prognosis [133]. In [42, 132], a greedy search is performed over a PIN network to identify a number of gene modules whose average expression is locally maximal.

The averaged expression values per module for each sample, called module activity, are used as features for a subsequent classification task. Discriminative power and correlation among genes in a linear path search is employed [119]. Linear paths are then combined to form modules, and module activity is inferred with a probabilistic method. In [134], a traditional support vector machine classifier is modified to consider the structure of gene interactions and force adjacent genes to contribute similarly when building the classifier. Minimal modules where numbers of differentially expressed member genes exceeded a pre-specified threshold are investigated in [135].

Each of those disease modules may reflect a specific functional pathway relevant to development of the disease of interest. However, it is argued that PIN data is generally unreliable and noisy, as PIN networks typically represent a collection of many interactions under various experimental conditions and cell cycle phases. Another problem of using PIN data is the high false positive rate, meaning that part of recorded interactions may not actually exist [136, 137], which in turn may lead to false discovery of biomarkers. Therefore, the adoption of canonical pathways, rather than protein interaction modules, using expertly curated knowledge may provide clearer insight into the interplay between genes in complex diseases.

3.1.3 Pathway-based Approaches

Biological pathways are a trusted expert-curated collection of molecular interaction networks. The availability of pathway information from publc databases, for example Kyoto Encyclopedia of Genes and Genomes (KEGG) [138], Gene Ontology (GO) [139] and Reactome [140], provide the possibility of analysing functional sets of genes that fall within common pathways and identifying the disease-relevant pathways as biomarkers. Initial efforts of gene set-based approaches included gene set enrichment analysis [141], which calculates to what extent a set of genes show statistically significant difference between samples belonging to either of two phenotypes. Other similar computational tools have also been reported [142–144]. However, those statistical frameworks commonly assign one score for each set of genes to quantify the deregulation of this gene set under disease status of interest, but does not provide more information on the level of gene set deregulation for each sample. It is argued that this drawback compromises their potential in personalised pathway analysis [40].

Therefore, a more informative approach may be to assign a score to each pathway and sample, which represents the activity of that particular pathway for that sample. The mean and median expression value across all constituent genes within a pathway, termed pathway activity, has been proposed in [145]. Other studies produce pathway activity measures based on principal component analysis (PCA) to derive the top principal component that captures the maximum variance in the dataset [40, 48, 146]. More recently a supervised greedy search algorithm was proposed that ranks genes according to their individual discriminative power and then searches for a subset of highly ranked genes whose averaged expression profiles yield better discriminative power [41]. This method was modified so that it accounts for up- and down-regulated genes by assigning positive sign and negative sign respectively [147]. Both methods are inherently applicable to binary classification problems. A statistical inference method [148] proposes to aggregate the probabilistic evidence of all genes within a pathway for predicting a sample into one of the two phenotypes. Other relevant studies based on the concept of pathway activity either require other biological information as prior, for example copy number variation and protein interactions [149] or are not designed for classification tasks [147, 150].

Pathway activity-based classification approaches provide competitive or higher prediction accuracy when compared to traditional single genes-based classifiers [41, 151], so extending or refining their use is a promising avenue for biomarker discovery. Despite rapidly increasing interest in developing novel and robust pathway activity inference methods, most of the existing methods still use rather simple means of summarising the expression patterns of either some or all constituent genes into the composite pathway level attribute, for example the mean or median value of sample expression across all or a subset of constituent genes [41, 145]. PCA-based methods [40, 48, 146, 152], calculate the first principal component, representing the maximum variance of the data set, as pathway activity. However such methods do not take into account the phenotype information of samples. Furthermore, some current pathway activity inference methods are constrained to two-phenotype (binary) classification problems [41, 147–149], disallowing their use in more complex problems of multi-phenotype classification.

In this work, we propose a novel multiclass method that infers pathway activity

in a supervised manner. The proposed method summarises expression patterns of constituent genes into pathway activity via weighted linear summation of gene expression. As opposed to some methods in literature where gene weights are taken as a prior, in our work gene weights are decided by the model, so that the constructed pathway activity can optimally distinguish samples from different phenotypes. Furthermore, the mathematical framework of this method offers the ability to the user to explicitly constraint the maximum number of constituent genes contributing to pathway activity inference. Using a number of published gene expression profile datasets, we show that this pathway activity inference method is robust in terms of the number of constituent genes allowed to determine the pathway activity metric. Comparative analyses show that the method is an effective means of reducing classification features, as it either outperforms or at least matches competing pathway activity inference methods in two-phenotype disease classification problems, and provides significantly better classification rates in multi-phenotype classification problems.

3.2 A Pathway Activity-based Disease Classification Procedure

An overview of the computational procedure developed for pathway activity-based disease classification is illustrated in Figure 3.1. A microarray gene expression profile and a set of pathways with their constituent genes form the input to create pathway-specific gene expression matrices. For each pathway, m denotes member genes, s samples and A_{sm} the expression value of gene m in sample s. To ensure gene expressions are of similar scales, data normalisation is performed on the raw data A_{sm}. More specifically, for each gene m, its expression values across all samples s are normalised to have mean of 0 and standard deviation of 1. This gives to the normalised expression data G_{sm}. The first stage of our computational procedure derives a new composite feature, pathway activity pa_s, from the pathway specific gene expression profile G_{sm}. The second stage of our protocol, the inferred pathway activities for all pathways are assembled to form a pathway activity profile matrix, on where a classifier is trained to predict the phenotype of a new sample. In next section, we present a novel mathematical model, which infers pathway activity with optimal classification accuracy.

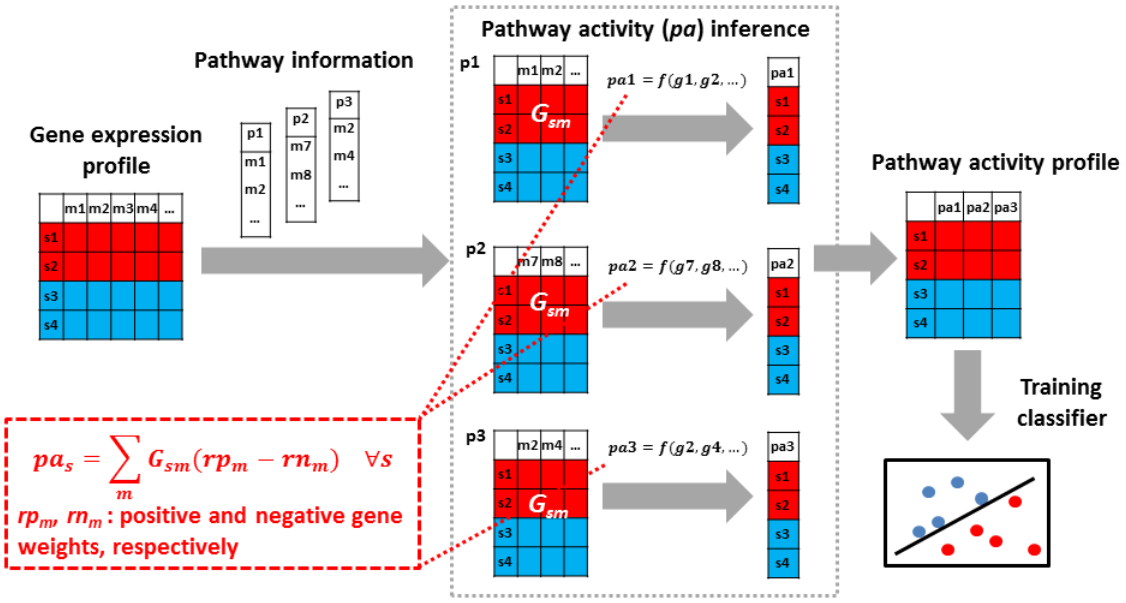

FIGURE 3.1: Schematic flow chart of the DIGS-based approach for multiclass disease classification problems.

3.2.1 A novel mathematical programming formulation to infer pathway activity

The indices, parameters and variables used in the model to infer pathway activity for each pathway are described below:

Indices	
s	sample
m	pathway member gene/feature
c,k	class label/phenotype
c_s	class label for sample s
Parameters	
A_{sm}	expression level of gene m on sample s
G_{sm}	standarised gene expression profile
U	an arbitrarily large positive number
ϵ	an arbitrarily small positive number
NoG	number of member genes allowed to have non-zero weight in building pathway activity for each pathway, a user-specific value
Positive variables	
rp_m	positive influence of gene m towards pathway activity inference
rn_m	negative influence of gene m towards pathway activity inference
Free variables	
pa_s	inferred pathway activity of sample s
LO_c	lower bound of range of class c on pathway activity
UP_c	upper bound of range of class c on pathway activity
Binary variables	
E_s	*1* if pathway activity of sample s falls within the range of its class; *0* otherwise
L_m	*1* if effect of gene m on pathway activity inference is positive; *0* if negative effect
Y_{kc}	*1* if upper bound of pathway activity range for class k is lower than lower bound of that for class c; *0* otherwise
W_m	*1* if gene m is active in pathway activity inference (have non-zero weight); *0* otherwise

Two sets of positive variables rp_m and rn_m are introduced, quantifying the positive

and negative weights of gene m towards pathway activity inference. For sample s, pathway activity, pa_s, is defined as the summation of the standardised gene expression values, G_{sm} multiplied by the gene weight $(rp_m - rn_m)$ over all member genes:

$$pa_s = \sum_m G_{sm}(rp_m - rn_m) \qquad \forall s \tag{3.1}$$

Both positive and negative weights of a gene m are defined as positive continuous variables, and their values are determined by the optimisation model. One set of binary variables, L_m, which takes values of either *0* or *1* has been introduced, while the following equations ensure that for each gene m at most one of rp_m and rn_m can take positive values:

$$rp_m = L_m \qquad \forall m \tag{3.2}$$

$$rn_m = 1 - L_m \qquad \forall m \tag{3.3}$$

When $L_m = 1$, rp_m can take any value between *0* and *1* while rn_m is forced to be equal to *0*; otherwise when $L_m = 0$, rp_m is forced to be equal to *0* while rn_m can be between *0* and *1*. In either case, both rp_m and rn_m can be equal to *0*, which means this particular gene has zero weight in inferring pathway activity. Overall, a gene can have positive, negative or zero weight towards the composite feature construction. For normalisation purpose, the summation of absolute gene weights should be equal to one:

$$\sum_m (rp_m + rn_m) = 1 \tag{3.4}$$

Inspired by [41], where a small subset of member genes is selected (usually less than 7) to construct pathway activity, we add constraints to limit the number of genes having non-zero weights in inferring pathway activity. Thus a new set of binary variables, W_m, are introduced to the model to indicate whether a member gene$m$ is active, i.e. having non-zero weights in constructing pathway activity or not:

$$rp_m + rn_m \leq W_m \qquad \forall m \tag{3.5}$$

If W_m takes the value of *0* then both positive weight (rp_m) and negative weight (rn_m) of gene m are forced to be equal to *0*, while when W_m is equal to *1*, gene m

is allowed to take any weight ($rp_m - rn_m$) between *-1* and *1*. The next equation restricts the maximum number of genes allowed to have non-zero weights to a manually specified value (NoG):

$$\sum_m W_m \leq NoG \tag{3.6}$$

In the case where NoG is equal to or larger than the number of member genes available in the pathway, the constraint is redundant as all the member genes will be allowed to take any weight ($rp_m - rn_m$) between *-1* and *1*. We aim to construct pathway activity as a feature with good discriminative power, which can separate samples from different phenotypes as much as possible.

For each phenotype/class c, two continuous variables have been introduced as LO_c and UP_c, denoting the lower and upper bound respectively, of the range of pathway activity for phenotype c. In addition, a set of binary variables, E_s, have been introduced, which is equal to *1* if activity value of sample s falls within the lower and upper bounds of its class range; *0* otherwise. The following constraints are also introduced:

$$0 \leq pa_s - LO_c + U(1 - E_s) \qquad \forall s, c_s \tag{3.7}$$

$$pa_s - UP_c - U(1 - E_s) \leq 0 \qquad \forall s, c_s \tag{3.8}$$

where c_s is the phenotype for sample s and U is an arbitrarily large positive constant. On the constructed pathway activity, ranges of different classes are not allowed to overlap. A set of binary variables, Y_{kc}, have been introduced as being equal to *1* if upper bound of range for class k is lower than lower bound of range for class c on pathway activity; *0* otherwise. The additional two sets of constraints have been introduced to guarantee the non-overlapping condition:

$$UP_k + \epsilon \leq LO_c + U(1 - Y_{kc}) \qquad \forall k < c \tag{3.9}$$

$$UP_c + \epsilon \leq LO_k + UY_{kc} \qquad \forall k < c \tag{3.10}$$

where ϵ is an arbitrarily small positive number ensuring that pair-wise classes do

not share a border. Equ. (3.9) and (3.10) are generated for each pair of classes. The objective of the optimisation problem is to infer the pathway activity such that it is as discriminative as possible, i.e. as many samples as possible can fall within the range of its corresponding classes ($E_s = 1$). In other words, the objective function is to minimise the number of misclassified samples:

$$min \quad \sum_{s}(1 - E_s) \tag{3.11}$$

The resulting mathematical programming-based formulation for inferring pathway activity is summarised below:

Objective function (3.11)
Subject to:
Pathway activity definition (3.1)
Positive and negative gene effect constraints (3.2) and (3.3)
Normalisation constraint (3.4)
Restriction of the number of active genes (3.5) (3.6)
Pathway activity enclosing constraints (3.7) and (3.8)
Non-overlapping constraints for ranges of different classes (3.9) and (3.10)
L_m, E_s, W_m, $Y_{kc} \in \{0, 1\}$; rp_m, $rn_m \geq 0$; pa_s, LO_c, UP_c: unrestricted

The proposed mathematical programming formulation consists of a linear objective function and a number of linear constraints. The linearity and presence of binary and continuous variables define an MILP model, named DIGS (DIfferential Gene Signatures) in this work, and can be solved to global optimality using some of the standard algorithms like branch-and-bound.

3.2.2 Comparison of the DIGS Model with Other Pathway Activity Inference Methods and Single Gene-based Methods

To compare the results obtained with the DIGS model, a number of pathway activity methods from the literature (summarised in Table 3.1) have been implemented. In overview, these methods include: i) the method that uses the microarray gene expression profile without pathway information, for example SG; ii) the method

that utilises pathway information but is based on the pathway specific gene expression profile instead of inferring pathway activity, for example Per_pathway [45], and iii) those that take advantage of pathway information and infer pathway activity, for example [41, 145, 146].

In detail, comparative results are presented by implementation of the following methods: i) a gene-based approach has been implemented for comparison where, given a whole gene expression profile, a feature selection [153] method is applied to select a subset of top genes with the best discriminative power for classification. The multiclass feature selection method [153] used here employs a distance metric, for example weighted L1 metric or *K-L* divergence and gives a subset of top attributes/genes with respect to the aggregated pair-wise class distances, where the number of attributes in the subset obtained is pre-set by the user. A classifier is then trained using only the small subset of discriminative genes for disease classification problems; ii) the Yang et al. [45] method, where each pathway-specific gene expression profile is treated independently, i.e. training and testing are conducted for each pathway-specific expression profile separately and classification accuracies across all pathways are averaged to obtain the final classification rate (referred as *per_pathway*), and iii) the two methods from Guo et al. [145] (referred as *mean* and *median*, respectively), which take either the mean or median gene expression values of all genes within a pathway for each sample. The Bild et al. [146] approach (referred as *PCA*) of using the first principal component as representation of pathway activity, which represents a family of principal component analysis-based methods. The Lee et al. [41] method, which works by identifying and averaging a subset of condition-responsive genes (referred as *CORGs*), has been implemented only for two-phenotype disease classification problems, as it is not suited to multi-class problems.

TABLE 3.1: Overview of evaluated methods

Guo et al. [145]	**Abbreviation**: Mean **Computational basis**: Pathway activity **Description**: Create pathway-specific gene expression profiles; for each pathway, pathway activity for sample is its mean expression value among all member genes; a classifier is trained on pathway activity profile.
Guo et al. [145]	**Abbreviation**: Median **Computational basis**: Pathway activity **Description**: Create pathway-specific gene expression profiles; for each pathway, pathway activity for sample is its median expression value among all member genes; a classifier is trained on pathway activity profile.
Bild et al. [146]	**Abbreviation**: PCA **Computational basis**: Pathway activity **Description**: Create pathway-specific gene expression profiles; for each pathway, top principal component is calculated as the pathway activity; a classifier is trained on pathway activity profile.
Lee et al. [41]	**Abbreviation**: CORGs **Computational basis**: Pathway activity **Description**: Create pathway-specific gene expression profiles; for each pathway, apply t-test to rank genes and perform a greedy search to find a subset of genes whose averaged expression values is locally maximal in t-test value; a classifier is trained on pathway activity profile; only applicable for two-class problems.
Yang et al. [45]	**Abbreviation**: Per_pathway **Computational basis**: Single genes **Description**: Create pathway-specific gene expression profiles; a classifier is trained on each pathway-specific gene expression profile separately, and prediction rates achieved by all pathway classifiers are averaged as the final prediction rate.
Single genes	**Abbreviation**: SG **Computational basis**: Single genes **Description**: Apply [153] to select a subset of top genes; a classifier is trained on reduced gene expression profile.
Proposed	**Abbreviation**: DIGS **Computational basis**: Pathway activity **Description**: Create pathway-specific gene expression profiles; Apply the proposed DIGS model to construct pathway activity as weighted linear summation of gene expressions; a classifier is trained on pathway activity profile.

3.3 Comparative Studies

A number of case studies have been employed to benchmark the performance of the proposed DIGS disease classification approach.

3.3.1 Data Sources

Complex diseases such as breast cancer and psoriasis are the product of multiple gene interactions that collectively contribute to the etiology of the disease through largely unknown mechanisms. Breast cancer is the most frequently diagnosed malignancy and has been intensively studied by gene expression profiling [6, 27, 31]. Psoriasis is a systemic, inflammatory skin disease with autoimmune underpinnings affecting 2-3% of the world population [154, 155]. Prostate tumor is the most frequently diagnosed cancer in American men [156] and displays a broad range of clinical and histological behaviors [113]. Diffuse large B-cell lymphoma (DLBCL) is the most common lymphoid malignancy in adults [33] with less than 40% patients responding desirably to the current therapy while the remainders succumb to the disease, highlighting the unidentified molecular heterogeneity in the tumors [157].

A total number of 8 published microarray gene expression profiles were obtained that represent these diseases (Table 3.2). In terms of disease phenotypes in these datasets, used as class outcomes in the relevant classification tasks, for psoriasis samples are either lesional or non-lesional tissue from psoriasis patients, as well as healthy controls [34, 158]. For breast cancer, 49 samples belong to three disease classes, apocrine, basal and luminal [159]; 139 samples are divided into healthy, luminal, ERBB2 and basal [27]; expression profiles of 230 breast cancer patients 48 of whom became residual invasive cancer free in the breast or lymph nodes after a 6-month preoperative chemotherapy and the remainder still had residual invasive cancer after the treatment. Gene expression data were generated using specimens of breast cancer before any treatment [115]; lymph-node negative breast cancer patients with some of them diagnosed with distant metastasis [25]. For prostate cancer, 102 expression profiles are used to distinguish tumour samples from normal samples [113]. Finally, 77 expression profiles of patients either diagnosed with diffuse large B-cell lymphoma (DLBCL) or follicular lymphoma (FL) are used [33].

All microarray datasets have been obtained on Affymetrix platforms. For each dataset, raw data have been downloaded and pre-processed using the Bioconductor package LIMMA [160]. KEGG C2 functional gene sets have been downloaded from MsigDB database (v3.0, Sep 2010) [161], which included a total number of 186 expert-curated pathways.

TABLE 3.2: Microarray gene expression datasets

Dataset	**Disease**	**# of samples**	**Phenotype**	**Source**
Swindell [158]	Psoriasis	180	Health control Psoriatic non-lesional skin Psoriatic lesional skin	GSE13355
Yao [34]	Psoriasis	82	Health control Psoriatic non-lesional skin Psoriatic lesional skin	GSE14905
Farmer [159]	Breast cancer	49	Apocrine tumour Basal tumour Luminal tumour	GSE1561
Pawitan [27]	Breast cancer	139	Normal Luminal tumour ERBB2 Basal	GSE1456
Singh [113]	Prostate cancer	102	Normal Tumour	broadinstitute.org
Shipp [33]	DLBCL	77	DLBCL FL	broadinstitute.org
Popovici [115]	Breast cancer	230	Residual invasive cancer No residual invasive cancer	GSE24061
Desmedt [25]	Breast cancer	198	Metastatic Non-metastatic	GSE7390

3.3.2 Evaluation of Classification Performance

The performances of various pathway activity metrics are evaluated by the classification accuracy achieved across the eight disease datasets. For each dataset, samples are split randomly in training and testing sets of 70% and 30% respectively and this procedure is repeated fifty times. Composite features are constructed using Mean, Median, CORGs, PCA and DIGS on the training samples, resulting in low dimensional matrix of samples across pathway activities, on which five popular classifiers SMO, NN, K-NN, Logistic Regression (Logistic) and HB are trained.

The classifiers are then tested on the testing sample set and the prediction accuracy is calculated as the number of correctly classified samples divided by the total number of testing samples, averaged across the fifty testing sets.

The above procedure is modified where pathway activities are not used, i.e. in the SG and per_pathway approaches. In the gene-based approach, the feature selection method [153] has been applied using training samples only and the top genes are selected. The number of top genes is set to be identical to the number of pathways (i.e. *186*) in order to derive comparable dimensionalities between the pathway activity-based and gene-based approach. For the *per_pathway* approach, each of the 5 classifiers have been trained using training samples only and then validated on the testing samples sets for each pathway separately.

Overall, 8 microarray gene expression profiles (DATASET), 7 competing methods (METHOD) and 5 classifiers (CLASSIFIER) are employed in our study. For each combination of DATASET, METHOD and CLASSIFIER, classification accuracies over 50 individual testing sets are averaged as the prediction accuracy for this combination. It is important to note that CORGs [41] is applicable for only two-phenotype problems, therefore we divide the 8 DATASETS into a group of 4 binary DATASETS and the other group of 4 multiclass DATASETS. For the binary classification comparison, for each METHOD we average the prediction accuracies over all 4 binary DATASETS and all 5 CLASSIFIERS, which gives a comprehensive indication of the efficiency of the evaluated METHODS (i.e. Mean, Median, PCA, CORGs, per_pathway, SG and the proposed DIGS). For the multiclass case, the same analysis is applied and all comparative analyses are discussed in the next section.

The DIGS model has been implemented in GAMS using the CPLEX MILP solver in a CentOS 5.2 64 bit Unix computer environment. The optimality gap is set as 0. Computational resource limit is set as 200 CPU seconds per run. Among the 5 classifiers SMO, NN, K-NN and Logistic have been implemented in WEKA, with the following parameters for NN: hidden layers 2, learning rate 0.1, momentum 0.2, training time 10000; and for K-NN: the number of nearest neighbours is selected as 5. For other classifiers, their default settings have been retained. HB has been reproduced in GAMS according to its original publication [74].

3.3.3 Sensitivity Analysis for *NoG*

Parameter *NoG* determines the maximum number of pathway member genes that have non-zero weight in activity inference. Tuning this parameter is important as a small value may not fully utilise the discriminative member genes, while an excessively large value may potentially cause over-fitting, i.e. in the case where too many genes are allowed to take non-zero weights for pathway activity against a relatively small number of training samples, leading to decreased prediction accuracy.

Here, the DIGS model is applied to infer pathway activity with *NoG* set to *5*, *10*, *15* and *20*, followed by training and testing using a range of classifiers for each microarray dataset. As a comparison, DIGS is also run with *NoG* set equal to the number of member genes for each pathway, so as to allow all member genes in a pathway to take non-zero weights for pathway activity inference. The prediction rates achieved by these different values of *NoG* are denoted by DIGS_5, DIGS_10, DIGS_15, DIGS_20 and DIGS_ALL and are shown in Figure 3.2 A and B with SMO and NN classifiers and other classifiers in Figures A.1, A.2 and A.3 in Appendix section.

Generally, the DIGS model is robust with respect to parameter *NoG*, as in the range of *5* to *20*, classification prediction performance is found to be mostly stable, with some improvement observed between *NoG=5* and *20*. Overall, it is noted that prediction performance is case-dependent, not only depending on the dataset under investigation, but also varying with the particular pathway in question (e.g. number of member genes per pathway). In some cases, some improvement is observed against the case of no selection, for example on Yao, Farmer and Pawitan datasets with SMO classifier classification rates increase from 83.7%, 88.3% and 92.9% to 89.5%, 97.6% and 98.8% (*NoG=5*), respectively (Figure 3.2 A).

The model performs well even in the case where the number of genes is not reduced (see DIGS_ALL in Figures 3.2 and A.1, A.2 and A.3), indicating that, although reducing the total number of genes per pathway through parameter *NoG* may be beneficial to a particular application, it is by no means compulsory. Therefore, *NoG* offers the flexibility of feature reduction, if looking into the effect of a subset of genes is desired, without imposing any additional limitations that would stem

from cases where parameter specification would be mandatory. For the implementations discussed below, *NoG* equal to a value of *10* was chosen as a sensible compromise of the effects discussed above.

3.3.4 Classification Rate Comparison against Other Methods

The performance of the proposed DIGS model against other competing methods in literature is compared and discussed here. As described in the previous section, extensive comparisons were implemented across 8 datasets (DATASET) and 7 competing methods (METHOD). To also account for the effect of classifier choice in the computational procedure, we tested the DIGS model across 5 classifiers (CLASSIFIER). The results across all DATASET, METHOD and CLASSIFIER combination are illustrated in Figures 3.3 A and B (for 5-NN and NN classifiers) and in the Appendix A.4, A.5 and A.6 (for SMO, HB and logistic).

It is obvious from Figure 3.3 A that using 5-NN as classifier DIGS-based classification approach achieves higher classification rates than other pathway activity inference methods, including Mean, Median, PCA and CORGs. On all 8 datasets, DIGS model inferring pathway activity has always outperformed other pathway activity inference methods. It is not a surprise as DIGS seeks to infer pathway activity as of optimal discriminative power. It is also true that DIGS-based pathway activity classification approach results in higher prediction accuracy than Per_pathway, where pathway-specific gene expression profiles are trained and tested independently without constructing pathway activity features. Lastly, the same observation can be made when comparing DIGS to SG, where 186 genes of best discriminative power are selected for classification. DIGS leads to better classification rates than SG on six occasions (Singh, Popovici, Desmedt, Swindell, Farmer and Pawitan), while being tied with SG on Yao and trailing SG by marginal extent on Shipp. Overall it is evident that the proposed DIGS-based classification approach leads to more robust and accurate classification than other state-of-the-arts approaches in literature.

With regards to the actual prediction rates, the combination of DIGS model inferring pathway activity and 5-NN classifier offers prediction rates of above 90% for 4 out of 8 employed datasets, including Singh, Shipp, Yao and Farmer, around

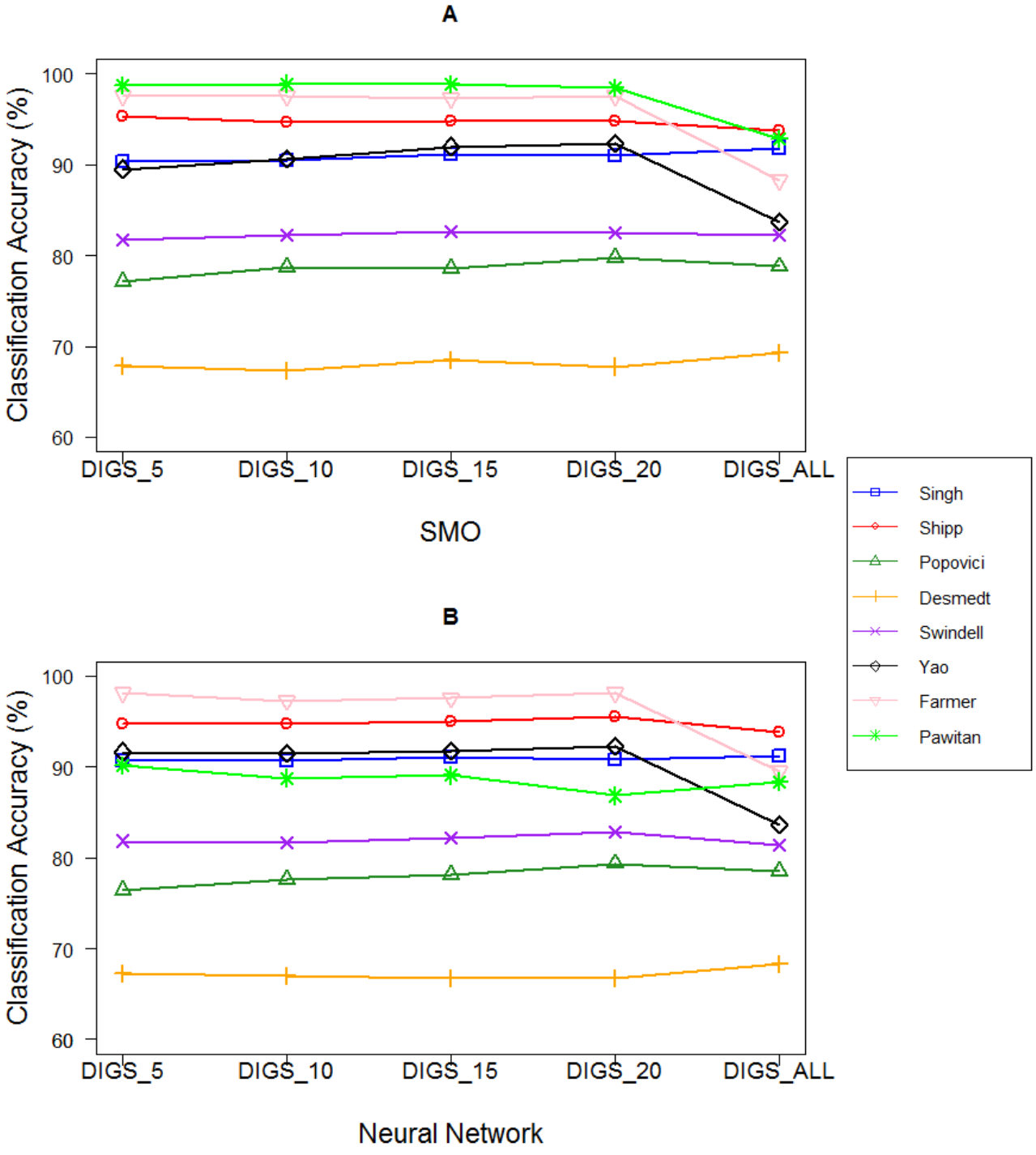

FIGURE 3.2: Sensitivity analysis of parameter *NoG* for DIGS model with SMO (A) and NN (B) classifiers. For each of the 8 datasets, the proposed DIGS model is applied to infer pathway activity while setting *NoG*, i.e. the maximum number of member genes in a pathway allowed to have non-zero weights, to *5*, *10*, *15* and *20*. In addition, DIGS model is also applied with *NoG* set to equal to the number of available member genes in a pathway, i.e. all member genes can take non-zero weights to construct pathway activity.

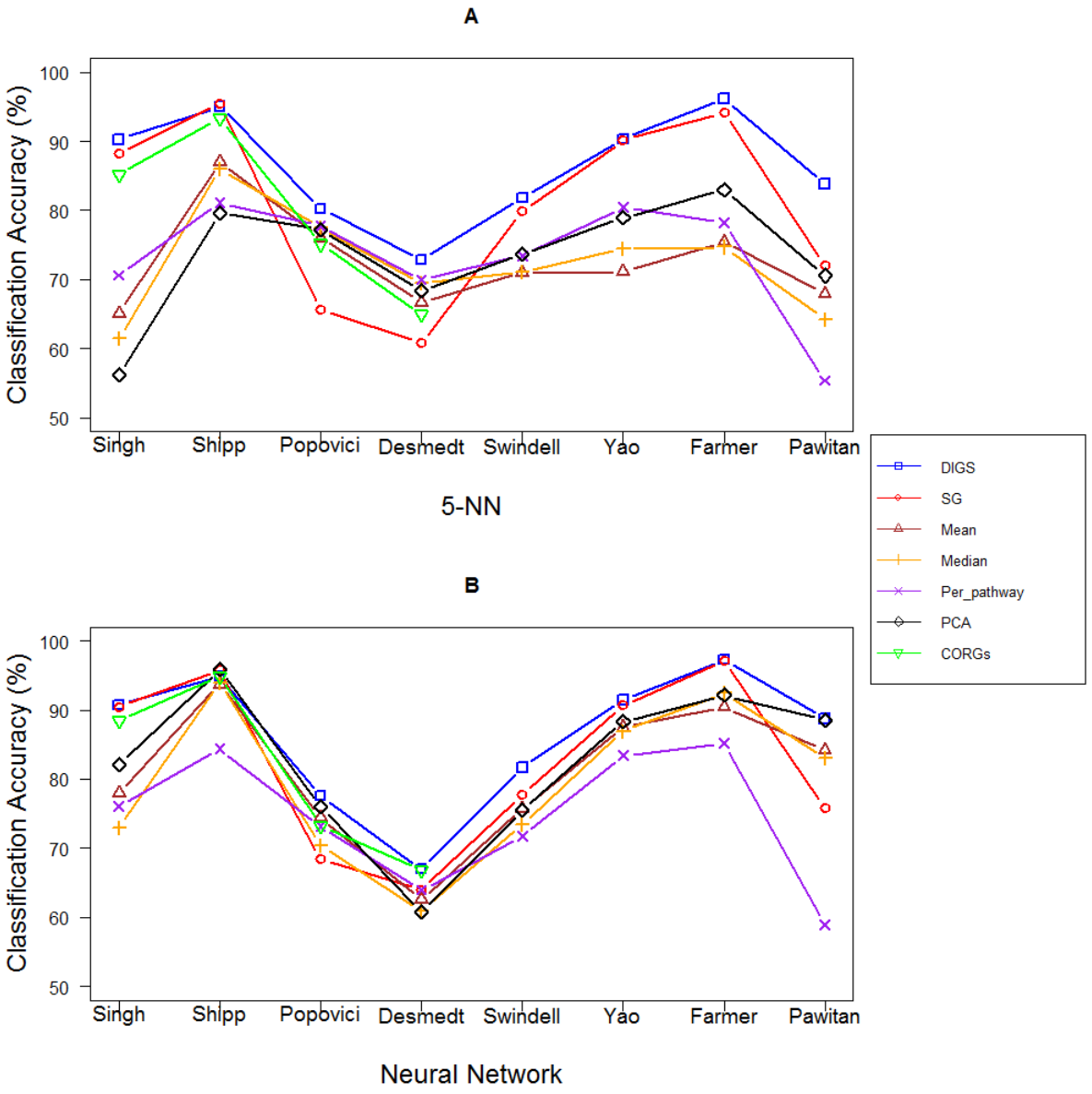

FIGURE 3.3: Classification accuracy comparison of 7 competing methods using 5-NN (A) and Neural Network (B) classifiers. The proposed DIGS pathway activity inference method is compared against other pathway activity inference methods (Mean, Median, PCA and CORGs) and also genes-based methods (SG and Per_pathway). Classification accuracy is summarised as average prediction rates over 50 runs of random partition of datasets into a 70% training set and a 30% testing set. With 5-NN classifier (A) and Neural Network classifier (B).

80% for another 3 datasets, including Popovici, Swindell and Pawitan, while still managed 70% for the last dataset Desmedt. The generally high prediction rates demonstrate the applicability and efficiency of the proposed DIGS model in practice.

To show that the desirable prediction rates achieved by DIGS-based approach is not due to a specific bias of DIGS model with 5-NN classifier, we present the classification accuracy comparison using Neural Network classifiers in Figure 3.3 B. According to Figure 3.3 B, when employing Neural Network classifier, DIGS-based disease classification approach again shows great competitiveness in 4 binary datasets that it gives the highest classification rate in Popovici; is tied as the top method in Singh with single genes-based approach and in Desmedt with CORGs; in Shipp DIGS trails the most accurate approach only marginally. In terms of 4 multiclass datasets, DIGS-based classification approach dominates in all of them. The same phenomenon can be observed using the other 3 implemented classifiers that DIGS model either provides competitive classification accuracies or gives the highest classification rate (See Figures A.4, A.5 and A.6 for more details).

To obtain an overview of how our methodology compares across all combinations of DATASET, METHOD and CLASSIFIER, we used a simple normalisation procedure where for each pair of DATASET and CLASSIFIER the actual prediction rates for every METHOD is divided by the highest prediction rates achieved throughout all METHODS. In other words, the normalised prediction rates, scaled between *0* and *1*, reflect the relative performance of a particular METHOD compared against the best performance across all methods for this specific combination of METHOD and CLASSIFIER. For example, on the Popovici dataset with 5-NN as classifier, the highest prediction rate across all 7 METHODS (achieved by DIGS as 80.14%) is given a score of *1* and for all other methods their prediction rates are divided with the highest prediction rate (in this case for DIGS), to express the relative performance of that method to the best, e.g. raw prediction accuracy of 75.13% achieved by CORGs is normalised to: $75.13\%/80.14\% = 0.9375$. For each combination of METHOD and CLASSIFIER, normalised prediction rates are averaged over 4 binary DATASETS and 4 multiclass DATASETS and are shown in Tables 3.3 and 3.4 respectively.

In terms of binary datasets, Table 3.3 clearly indicates that DIGS pathway inference model comes at the top of all METHODS. This is true in the case of most classifiers used and it is only when using with logistic as classifier where DIGS is outperformed by CORGs and SG. For multi-class datasets (Table 3.4) DIGS is the best method throughout, indicating the strength of our proposed methodology for the most challenging cases where multiple outcomes need to be predicted. This

highlights that one of the contributions of this work is to design, according to the authors' best knowledge, the first supervised pathway activity inference method applicable to both binary and multiclass datasets.

TABLE 3.3: Mean normalised classification rates over 4 two-phenotype datasets

two-class	5-NN	NN	SMO	HB	Logistic	Average
DIGS	**0.9988**	**0.9973**	**0.9757**	**0.9835**	0.9318	**0.9774**
SG	0.9071	0.9584	0.9474	0.9730	**0.9816**	0.9535
Mean [145]	0.8737	0.9323	0.9435	0.8819	0.8902	0.9043
Median [145]	0.8751	0.9004	0.9225	0.8707	0.8789	0.8895
Per_pathway [45]	0.8903	0.9041	0.9325	0.8547	0.8632	0.8890
PCA [146]	0.8389	0.9480	0.9704	0.8402	0.8482	0.8891
CORGs [41]	0.9371	0.9769	0.9645	0.9595	0.9684	0.9613

*The highest normalised mean accuracy for each classifier is highlighted in bold, same for Table 3.4

TABLE 3.4: Mean normalised classification rates over 4 multi-phenotype datasets

two-class	5-NN	NN	SMO	HB	Logistic	Average
DIGS	**1**	**1**	**1**	**1**	**1**	**1**
SG	0.9532	0.9488	0.9335	0.9241	0.8290	0.91772
Mean [145]	0.8126	0.9402	0.9372	0.7518	0.5614	0.80064
Median [145]	0.8090	0.9334	0.9246	0.7639	0.5440	0.79498
Per_pathway [45]	0.8158	0.8322	0.8521	0.7893	0.5589	0.76966
PCA [146]	0.8696	0.9585	0.9452	0.8043	0.6450	0.84452

3.3.5 DIGS identifies disease relevant pathways

Besides the high classification rates achieved by the proposed DIGS model, we have also identified a number of breast cancer pathways that may indicate pathway biomarkers. For Pawitan, where around 90% classification rates can be achieved using DIGS with all 5 classifiers, we employed an information gain feature ranking method in WEKA to rank the constructed pathway activities for each random training set. We record 11 pathways that are ranked more than *20* times as the most discriminative. As we have constrained the proposed DIGS model to allow

only *10* genes per pathway to participant in pathway activity inference, we further extract for each identified significant pathway the set of constituent genes included in the active genes more than *10* times.

The set of pathways and genes that are found as most discriminant with our method are listed in the Table 3.5 below. Apart from obvious links to cancer pathways, such as prostate cancer, and other well-known signalling pathways that are known to be deregulated in tumorigenesis (Wnt signalling [162, 163]), we note deregulation of nitrogen metabolism that has recently been linked to breast cancer [164]. Ubiquitin-mediated proteolysis is also identified, in accordance to previous reports about the importance of this pathway in disease [165] and is linked to poor survival in breast cancer [166]. Glycosylation is also known to be altered in cancer cells where overexpression of large glycoproteins such as mucins has been characterized [167]. Enzymes from the family of GALNT6 and GALNT14 that we have identified were found to be elevated in breast and gastric carcinomas [168]. We also identify the adherens junction complex, that comprises of cadherins and the catenins, is a major adhesion structure in endothelial cells and has been implicated in playing a fundamental role in controlling the transport across the endothelial barrier and in regulating angiogenesis [169] and has been shown to be affected in invasive breast cancer [170].

We also draw pathway activity heat maps for the significant pathways identified in Pawitan. In Figure 3.4, pathway activities are inferred using all samples. Pathways are clustered based on similarity of activities on Euclidean distance. It is clear from Figure 3.4 that pathways are divided into two main clusters, showing distinct patterns of expression. Ubiquitin mediated proteolysis pathway, Erbb signalling pathway, O glycan biosynthesis pathway, Dorso ventral axis formation pathway and prostate cancer pathway are shown to be associated with up-regulation in Luminal tumour, and down-regulation in Basal tumour. The other significant pathways appear to have the opposite regulation mechanism, i.e. they are down-regulated in Luminal tumour and up-regulated in Basal tumours.

Table 3.5: Significant pathways and constituent genes identified by the proposed DIGS model for Pawitan

Pathway	**Significant constituent genes**
PROSTATE CANCER	EGFR, TCF7L1, GSTP1, PDGFRA, CCNE1, CHUK, PIK3R3, ERBB2, PIK3R1
UBIQUITIN MEDIATED PROTEOLYSIS	UBE2E3, MID1, SKP2, BRCA1, WWP1
WNT SIGNALING PATHWAY	FZD7, SOX17, TCF7L1, SKP1, SFRP1, FZD8
O GLYCAN BIOSYNTHESIS	GALNT3, GALNT7, GALNT11, GALNT6, GCNT3, B4GALT5, GALNT8, C1GALT1, GALNT12, GCNT4, GALNT14, GALNT10, GALNT2, ST3GAL2, GCNT1, ST3GAL1, C1GALT1C1, GALNT1
ADHERENS JUNCTION	EGFR, ERBB2, TCF7L1, TCF7L2, MET, RAC3, SMAD3, MLLT4, RHOA
ERBB SIGNALING PATHWAY	EGFR, NCK2, ERBB2, AKT3, PAK4, EREG, MAPK9, AKT2
NITROGEN METABOLISM	CA12, CA5A, CA9, GLUL, CA3, CA14, CA8, CA7, CA5B, GLUD1, CA2, AMT, CA6, CA1, CTH, GLS2, GLUD2, HAL, CA4, ASNS, CPS1
DORSO VENTRAL AXIS FORMATION	EGFR, NOTCH1, GRB2, MAPK3, NOTCH3, SOS1, CPEB1, PIWIL2 ETS2, MAPK1, NOTCH4, ETV6, PIWIL1, MAP2K1, NOTCH2, SOS2, ETS1, ETV7, KRAS
ENDOMETRIAL CANCER	EGFR, TCF7L1, ERBB2, TCF7L2, MLH1, ELK1, NRAS, AKT3, ARAF, CTNNA2, PIK3CB, AKT2, CCND1, FOXO3, LEF1
NON SMALL CELL LUNG CANCER	EGFR, AKT3, E2F3, ERBB2, BAD, E2F1, RARB, CDKN2A, PLCG2, GRB2, HRAS, MAPK3, PIK3CD, RXRG, TGFA
PANCREATIC CANCER	EGFR, ERBB2, AKT3, CDKN2A, MAPK9, PLD1, RAC3, RALA, CCND1, E2F3, JAK1, PIK3R1

3.4 Concluding Remarks

Incorporating pathway information as biological priori with microarray gene expression profile has been demonstrated to be a promising alternative to conventional gene-based approach in various disease classification problems. However to the author's best knowledge there are no supervised pathway activity inference methods for multiclass disease classification problems. In this work, a novel supervised pathway activity inference method for both binary and multiclass disease classification problems, DIGS, has been proposed using mathematical programming optimisation techniques. For each pathway, a new composite feature, called pathway activity, is constructed as a weighted linear summation of expressions of member genes. In each pathway the number of member genes contributing to pathway activity inference by taking non-zero weights is constrained explicitly. The proposed DIGS model provide three main benefits over the existing pathway

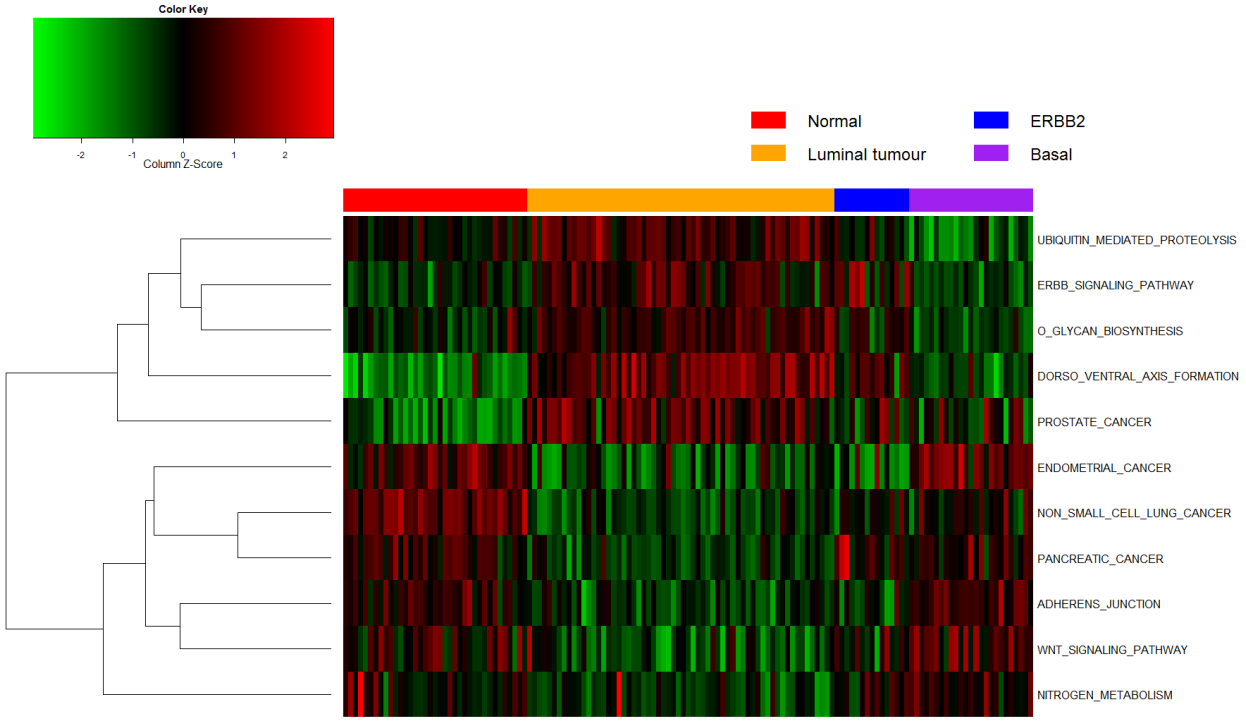

FIGURE 3.4: Pathway activity of the significant pathways in Pawitan. Pathway activities are inferred with DIGS model using all samples. Red/green blocks indicate up-/down- regulation of pathways (rows) in samples (columns). Pathways are clustered according to similarity of their activities.

activity inference methods in literature: (a) the weights of constituent genes in building pathway activity are optimised by DIGS in order to maximise the discriminative power of the pathway activity; (b) the maximum number of constituent genes taking non-zero weights when building pathway activity can be explicitly specified by user; (c) the proposed pathway activity inference model is applicable to both binary and multiclass disease classification problems.

A total number of 8 microarray gene expression profiles totalling 877 samples and 100,000 genes have been used to demonstrate the applicability and efficiency of the proposed pathway activity inference scheme. The classification results show that for 4 two-class problems DIGS-based classification approaches lead to higher normalised classification performance compared to other existing pathway-based approaches as well as genes-based approaches. In terms of multiclass classification problems, mathematical programming inferring pathway activity here gives consistently the highest prediction accuracies that with the same classifier DIGS always outperforms others by distance.

One shortcoming of our previous pathway activity inference method 27 is its computational complexity. Due to the combinatorial nature of inferring pathway activity, the required computational time grows exponentially as the number of samples and pathway constituent genes. It is generally infeasible to achieve globally optimal solution for microarray datasets of moderate to large sizes. In terms of future work, we plan to propose computationally more efficient alternative optimisation models via replacing much of the difficulty binary variables with continuous variables. Another possible future direction is to accommodate non-linearity in the pathway activity inference. The existing activity inference methods in literature almost all assume a simple linear algebraic relationship between phenotypic outcome and constituent genes. A non-linear pathway activity inference method would be more flexible and powerful to model complex gene-phenotype relationships. One possible means to achieve this can be introduction of new pseudo genes as higher order polynomials and pair-wise products of the original genes before optimising the weights for both original and pseudo genes. To ensure computational efficiency and reliability, a gene selection method, for example t test or information gain, should be implemented to select only a small subset of highly differentially expressed original genes, which serve as the basis of creating pseudo genes.

Chapter 4

A Novel Piece-wise Linear Regression Model

Data regression aims to address the problem of predicting continuous output variables from several independent input variables by approximating their relationship. In this chapter, a novel piece-wise linear regression model is presented, based on mathematical programming optimisation techniques. The proposed method separates samples into multiple regions by segmenting one input feature, and fits one *distinct* linear regression function per region to minimise the training error. An efficient solution algorithm has also been proposed that identifies the key partitioning feature and the number of regions.

4.1 Introduction and Literature Review

In data mining, regression is a type of analysis that predicts continuous output/response variables from several independent input variables. Given a number of samples, each one of which is characterised by certain measurable input and output variables, regression analysis aims to approximate their functional relationship. The estimated functional relationship can then be used to predict the level of output variables for new enquiry samples. Generally, regression analysis can be useful under two circumstances: 1) when the process of interest is a black-box, i.e. the knowledge of the underlying mechanism of the system is incomplete. In this case, regression analysis can make predictions on the level of output variables

from the relevant input variables without requiring details of the however complicated inner mechanism [171–174]. Quite frequently, the user would also like to gain some valuable insights into the true underlying functional relationship, which means the interpretability of a regression method is also of importance, 2) when the detailed simulation model relating input variables to output variables, usually via some other intermediate variables, is known, yet is too complex and expensive to be evaluated comprehensively in feasible computational time. In this case, regression analysis is employed to approximate the overall system behaviour with much simpler functions while preserving a desired level of accuracy, and can then be more cheaply evaluated [175–179].

A large number of regression analysis methodologies exist in the literature, including: linear regression, support vector regression (SVR), kriging, radial basis function (RBF) [180], multivariate adaptive regression splines (MARS), multilayer perceptron (MLP), random forest, K-nearest neighbour (KNN) and piecewise regressions. Those regression methodologies are briefly summarised before presenting our proposed method.

4.1.1 Linear Regression

Linear regression is one of the most classic types of regression analysis, which predicts the output variables as linear combinations of the input variables. The regression coefficients of the input variables are usually estimated using least squared error or least absolute error approaches, and the problems can be formulated as either quadratic programming or linear programming problems, which can be solved efficiently. In some cases when the estimated linear relationship fails to adequately describe the data, a variant of linear regression analysis, called polynomial regression, can be adopted to accommodate non-linearity [181]. In polynomial regression, higher degree polynomials of the original independent input variables are added as new input variables into the regression function, before estimating the coefficients of the aggregated regression function. Polynomial functions of second-degree have been most frequently used in literature due to its robust performance and computational efficiency [182].

Another popular variant of linear regression is called least absolute shrinkage and selection operator (LASSO) [183]. In LASSO, summation of absolute values of

regression coefficients is added as a penalty term into the objective function. The nature of LASSO encourages some coefficients to equal to *0*, thus performing implicit feature selection [184].

Automated learning of algebraic models for optimisation (ALAMO) [185, 186] is a mathematical programming-based regression method that proposes low-complexity functions to predict output variables. Given the independent input features, ALAMO starts with defining a large set of potential basis functions, such as polynomial, multinomial, exponential and logarithmic forms of the original input variables. Subsequently an mixed integer linear programming model is solved to select the best subset of basis functions that optimally fit the data. The cardinality of the subset is initially set equal to *1* and then iteratively increased until the Akaike information criterion, which measures the generalisation of the constructed model, starts to increase [187]. The model aims to capture the synthetic effect of different basis functions, which is considered more efficient than traditional step-wise feature selection. Note that the combinational nature of the integer programming model poses great computational difficulty for large-size datasets, making it hard to identify quality solutions.

4.1.2 SVR

Support vector machine is a very established statistical learning algorithm, which fits a hyper plane to the data in hand [188]. SVR minimises two terms in the objective function, one of which is ϵ-insensitive loss function, i.e. only sample training error greater than an user-specific threshold, ϵ, is considered in the loss function. The other term is model complexity, which is expressed as sum of squared regression coefficients. Controlling model complexity usually ensures the model generalisation, i.e. high prediction accuracy in testing samples. Another user-specified trade-off parameter balances the significance of the two terms [189]. One of the most important features that contribute to the competitiveness of SVR is the kernel trick. Kernel trick maps the dataset from the original space to higher-dimensional inner product space, at where a linear regression is equivalent to an non-linear regression function in the original space. A number of kernel functions can be employed, e.g. polynomial function, radial basis function and fourier series. Formulated as a convex quadratic programming problem, SVR can be solved to global optimality.

Despite the simplicity and optimality of SVR, the problem of tuning two parameters, i.e. training error tolerance ϵ and trade-off parameter balancing model complexity and accuracy, and selection of suitable kernels still considerably affect its performance accuracy [190, 191].

4.1.3 Kriging

Kriging is a spatial interpolation-based regression analysis methodology [192]. Given a testing sample, kriging estimates its output as a weighted sum of the outputs of the nearby training samples. The weights of samples are computed solely from the data by considering sample closeness and redundancy, instead of being given by an arbitrary decreasing function of distance [193]. The interpolation nature of kriging means that the derived interpolant passes through the given training data points, i.e. the error between predicted output and real output is zero for all training samples. Different variants of kriging have been developed in literature, including the most popular ordinary kriging [194] and universal kriging [195].

4.1.4 MARS

MARS [18] is another type of regression analysis that accommodates non-linearity and interaction between independent input variables in its functional relationship. Non-linearity is introduced into MARS in the form of the so-called hinge functions, which are expressions with max operators and look like $max(0, X - const)$. If independent variable X is greater than a constant number *const*, the hinge function is equal to *X-const*, otherwise the hinge function equals to *0*. The hinge functions create knots in the prediction surface of MARS. The functional form of MARS can be a weighted sum of constant, hinge functions and products of multiple hinge functions, which makes it suitable to model a wide range of non-linearity.

The building of MARS usually consists of two steps, a forward addition and a backward deletion step. In the forward addition step, MARS starts from one single intercept term/constant and iteratively adds pairs of hinge functions (i.e.

$max(0, X - const)$ and $max(0, const - X)$) that leads to largest reduction in training error. Afterwards, a backward deletion step, which removes one by one those hinge functions contributing insignificantly to the model accuracy, is employed to improve generalisation of the final model [196]. The presence of hinge functions also make MARS a piece-wise regression method.

4.1.5 MLP

Multilayer perceptron is a feedforward artificial neural network, whose structure is inspired by the organisations of biological neural networks [197]. A MLP typically consists of an input layer of measurable features, an output layer of response variables, sandwiching multiple intermediate layers of neurons. The network is fully interconnected in the sense that neurons in each layer are connected to all the neurons in the two neighbour layers. Each neuron in the intermediate layers takes a weighted linear combination of outputs from all neurons in the previous layer as input, applies an non-linear transformation function before supplying the output to all neurons of the next layer. The use of non-linear transformation functions, including sigmoid, hyperbolic tangent and logarithmic functions, makes MLP suitable for modelling highly non-linear relationship [198].

Identifying the optimal configuration of a MLP, i.e. the number of intermediate layers, the number of neurons for each intermediate layer, the type of activation function for each neuron and the weights of connection between consecutive layers of neurons, is known to be time-consuming and traps in local optimal solutions [199]. The large degree of freedom in training a MLP is often blamed for data over-fitting. The architecture of a MLP is almost always fixed by the user and back-propagation is used to tune only the weights of connection between neighbour layers of neurons [200].

4.1.6 Random Forest

Before introducing random forest we first describe *regression tree*, which is a decision tree-based prediction model. Starting from the entire set of samples, a regression tree selects one independent input variable among all and performs binary split into two child sets, under the condition that the two child nodes give

increased purity of the data compared with its single parent node. Purity is often defined as the deviation of predicting with the mean value of the output variable. The process of binary split is recursively applied for each child node until a terminating criterion is satisfied. The nodes that are not further partitioned are called terminal leaves. After growing a large tree, a pruning process is employed to remove the leaves contributing insignificantly to the purity improvement [19, 201]. In order to improve model fit, a linear regression model can be fitted for each leaf [202].

Random forest is an ensemble learning method of regression trees. In general, random forest [203, 204] builds a forest of multiple regression tree models and aggregate the decisions from all the trees to produce a final prediction. Given a dataset, multiple bootstrap sample sets are first created by random sampling with replacement. Each of the bootstrap sample set is then learned by a revised regression tree algorithm, which differs from the classic regression tree by randomly selecting a candidate subset of features for each binary split of node [205]. The accuracy of each regression tree can be estimated on the training samples absent from the bootstrap set, and the final prediction can be either simple average of predictions from all trees or weighted average considering the estimated accuracy. It is demonstrated that random forest achieves more robust prediction performance compared with single regression tree method [206].

4.1.7 KNN

KNN belongs to the category of lazy learning algorithms, due to the fact that prediction is based on the available instances without an explicit training phase of constructing mathematical models, thus making it one of the simplest regression methods in literature [207]. Given a testing sample, KNN first identifies K closest instances in the training sample set, the exact value of K is given *a priori*. The closeness of samples can be measured by different distance metrics, for example Euclidean and Manhattan distances. Prediction is then taken as weighted mean of the outputs of the K nearest neighbours, with weight often being defined as the inverse of distance [208]. Despite its simplicity, KNN usually provides competitive prediction performance against much more sophisticated algorithms.

4.1.8 Previous Work on Piece-wise Regression

Piece-wise functions have been frequently studied in literature as well. In [209], univariate piece-wise linear functions have been used to fit ecological data and identify break-points that represent critical threshold values of a phenomenon. In [210], a method based on statistical testing is proposed to estimate the number of break-points for an univariate piece-wise linear function. Malash & El-Khaiary [211] also apply piece-wise linear regression techniques on univariate experimental adsorption data. Piece-wise function is determined by solving a non-linear programming model.

SegReg (www.waterlog.info/segreg.htm) is a free software that permits estimating of piece-wise regression functions with up to two independent variables. For one independent variable, SegReg splits from a series of candidate break-points and for each one fits a linear regression for either side of the break-point. The break-point corresponding to the largest statistical confidence is taken as the final solution. In the case of two independent variables, SegReg first determines the two-region piece-wise regression function between the dependent variable and the most significant input variable, before computing the relation between its residual/deviation and the second input variable. Segmented [212] is a package written in R [213], which also outputs a simple form of piece-wise linear regression functions. Segmented requires a user to specify the segmented input variables, the number of break-points and also the initial guess of each break-point. Starting from the those supplied initial positions of break-points, Segmented iteratively searches around the neighbour of the initial guess points to identify break-points of the best quality [214]. However, in practice, it is difficult to reasonably supply good starting points especially for multivariate datasets, where visual examination is extremely difficult if not impossible. This limitation makes it hard to identify quality solutions. On the other hand, in Segmented, only the input variables being segmented can have different regression coefficients across different segments, while the other input variables keep the same coefficients across the whole range.

Both Magnani & Boyd [215] and Toriello & Vielma [216] publish work on data fitting with a special family of piece-wise regression functions, called max-affine functions. The form of max-affine functions is defined as the maximum of a series of linear functions, i.e. a sample is projected to all linear functions, and the

maximum projected value among all is taken as final predicted value from the piece-wise functions. The use of max-affine functions limits the fitted surface to be convex. In [215], a heuristic method is used to ease the difficulty of direct solving the highly non-linear max-affine functions, while in [216], big-M constraint is used to reformulate the problem into an non-convex mixed integer non-linear programming model. However, computational complexity is limiting their applications to examples of small scale.

In this work, we propose a novel piece-wise regression method for multivariate regression problems using mathematical programming optimisation techniques. A single input variable is partitioned to separate samples into multiple regions, while each region is fitted with a unique linear regression function. It is first assumed that both the partitioning feature and the number of break-points are known. Under this assumption, we propose an optimisation model that optimally estimates the position of all break-points and the linear regression coefficients for each region *simultaneously* so that the total absolute deviation is minimised. Furthermore, a solution procedure is used to identify the key partitioning feature and the number of break-points. A number of multivariate benchmark datasets have been used to demonstrate the applicability and efficiency of the proposed regression method.

4.2 A Novel Piece-wise Linear Regression Method

A novel piecewise linear regression method is proposed in this work. The core idea of the proposed method is to identify a single input feature, and separate the samples into complementary regions on this feature. One *unique* linear regression function is fitted for each local region. The sample partition and calculation of local regression coefficients are performed *simultaneously* within the proposed optimisation to achieve least absolute error.

4.2.1 A Novel Regression Method

In this section, we first describe a novel mathematical programming model that optimises the location of break-points and regression coefficients for each region so as to achieve minimal training error, given as prior the key partitioning feature and the total number of regions. Subsequently, a solution procedure is proposed

to identify the best partition feature and the number of regions.

The indices, parameters and variables associated with the proposed model are listed below:

Indices	
s	sample, *s=1,2,...,S*
m	feature/independent input variable, *m=1,2,...,M*
r	region, *r=1,2,...,R*
m^*	the single feature where segmentation takes place
Parameters	
A_{sm}	numeric value of sample s on feature m
Y_s	real output value of sample s
U', U"	arbitrarily large positive numbers
Free variables	
W_m^r	regression coefficient for feature m in region r
B^r	intercept of regression function in region r
$Pred_s^r$	predicted output for sample s in region r
$X_{m^*}^r$	break-point r on partition feature m^*
D_s	training error between predicted and real outputs for sample s
Binary variables	
F_s^r	*1* if sample s falls into region r;
	0 otherwise

Assume first that both the partitioning feature m^* and the number of regions R are given, the *R-1* break points are arranged in an ordered way:

$$X_m^{r-1} \leq X_m^r \qquad \forall m = m^*, r = 2, 3, ..., R \tag{4.1}$$

Binary variables F_s^r are introduced to model if sample s belongs to region r or not. Modelling of which sample belongs to which region is achieved with the following constraints:

$$X_m^{r-1} - U'(1 - F_s^r) \leq A_{sm} \qquad \forall s, r = 2, 3, ..., R, m = m^* \tag{4.2}$$

$$A_{sm} \leq X_m^r + U'(1 - F_s^r) \qquad \forall s, r = 1, 2, ..., R-1, m = m^* \tag{4.3}$$

When sample s belongs to region r (i.e. $F_s^r = 1$), A_{sm^*} falls into the region bounded by the two consecutive break-points $X_{m^*}^{r-1}$ and $X_{m^*}^{r}$ on feature m^*; otherwise the two sets of constraints become redundant. A visualisation of break-points and regions is provided in Figure 4.1:

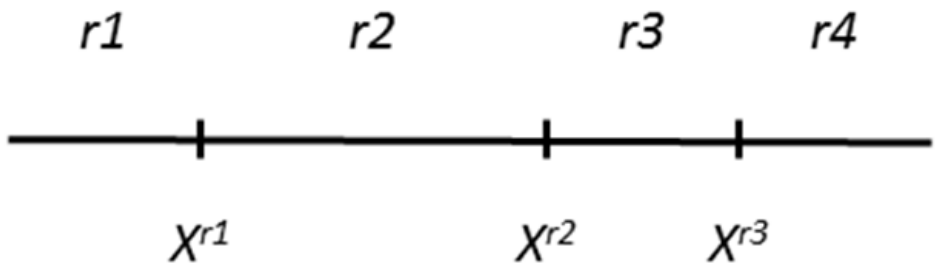

FIGURE 4.1: Break-points and regions. On the key partitioning feature m^*, break-points are arranged so that $X^{r1} < X^{r2} < X^{r3}$ ....

The following constraints restrict that each sample belongs to one and only one region:

$$\sum_r F_s^r = 1 \qquad \forall s \tag{4.4}$$

For sample s, its predicted output value for region r, $Pred_s^r$, is as below:

$$Pred_s^r = \sum_m A_{sm} W_m^r + B^r \qquad \forall s, r \tag{4.5}$$

For any sample s, its training error/residual is equal to the absolute deviation between the real output and the predicted output for the region r where it belongs to (i.e. $F_s^r = 1$):

$$D_s \geq Y_s - Pred_s^r - U''(1 - F_s^r) \qquad \forall s, r \tag{4.6}$$

$$D_s \geq Pred_s^r - Y_s - U''(1 - F_s^r) \qquad \forall s, r \tag{4.7}$$

The objective function is to minimise the sum of absolute training error:

$$min \sum_s D_s \tag{4.8}$$

The final model, named as Optimal Piece-wise Linear Regression Analysis (OPLRA) in this work, is summarised as below:

Objective function (4.8)
Subject to:
Positions of break-points (4.1)
Sample enclosing constraints (4.2) and (4.3)
One-region constraints (4.4)
Sample predicted output values (4.5)
Sample residual (4.6) and (4.7)
$D_s \geq 0$, $F_s^r \in \{0,1\}$, W_m^r, B^r, $Pred_s^r$, $X_{m^*}^r$: unrestricted

OPLRA consists of a linear objective function and several linear constraints, and the presence of both binary and continuous variables define an MILP problem. A heuristic solution procedure is also proposed in this work to identify the partitioning feature (m^*) and the number of regions (R), as described in Figure 4.2 below.

The heuristic procedure starts with solving a multivariate linear regression on the entire set of training data with least absolute deviation. Subsequently, each input feature in turn serves as partition feature m^* once and the OPLRA model is solved while allowing two regions (i.e. $R = 2$). The feature corresponding to the minimum training error is kept and if its error represents a percentage reduction of more than β from the global linear regression without data partition ($\frac{Error_{R=1} - Error_{R=2}}{Error_{R=1}} \geq \beta$), the procedure continues; otherwise it is decided that two-region piecewise linear regression does not provide a desirable improvement upon the classic linear regression model, and the initially derived linear regression function without sample partition is obtained for prediction. The parameter β, taking value between *0* and *1*, quantifies the percentage reduction in training error that justifies adding one more region. If two-region piecewise regression is accepted, the corresponding partition feature is retained for further analysis while the number of regions is iteratively increased, until the β training reduction criterion is not satisfied between iterations.

β is the only user-specific parameter in our proposed regression method, which requires fine tuning. A small value may cause over-fitting, i.e. too many regions

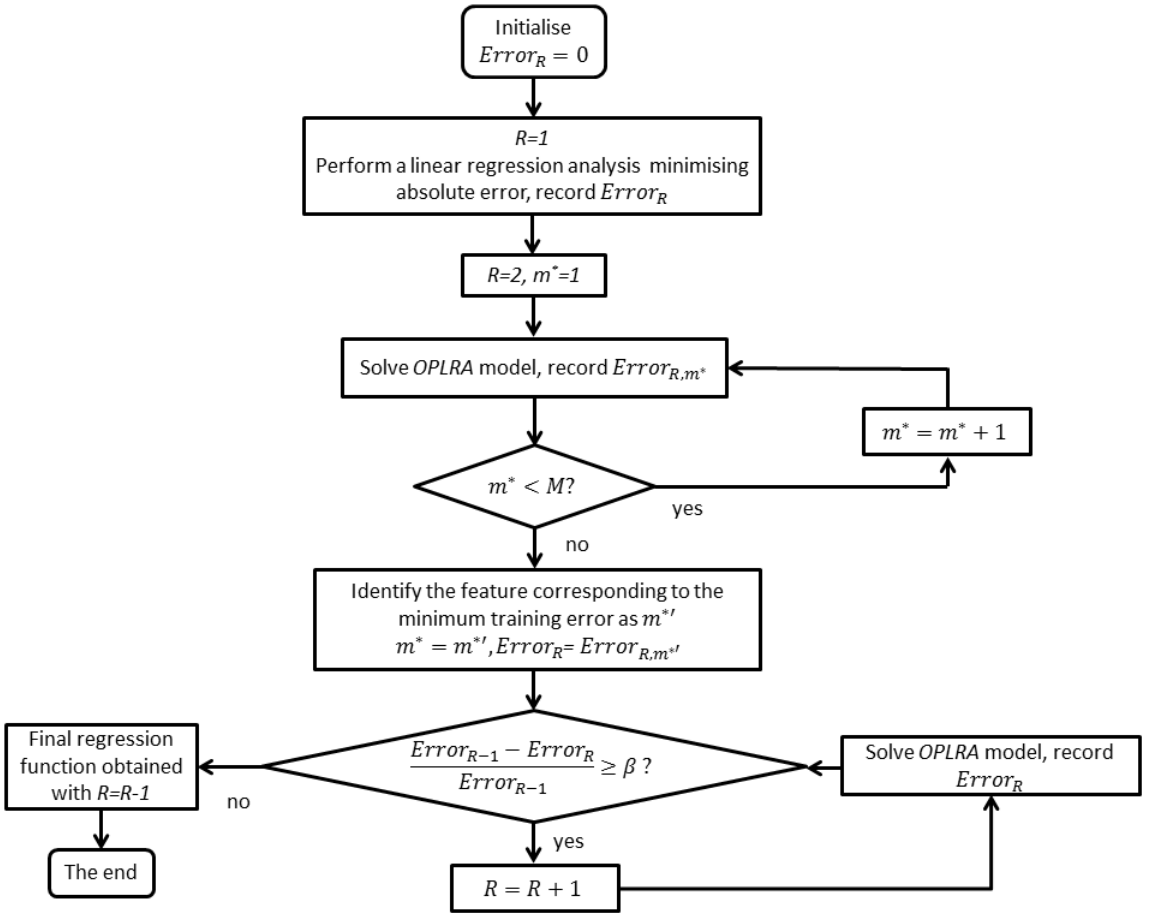

FIGURE 4.2: A heuristic procedure to identify the partition feature and the number of regions.

are allowed and each region contains only small number of samples, which then results in unreliable construction of local linear regression functions; while a value excessively large will lead to premature termination of growing regions, which then under-fit the data. In the next section, a series of values is tested on a number of benchmark datasets and the optimal value corresponding to the most robust prediction performance can be easily identified.

The constructed piecewise linear regression functions are then used to predict the output value of new samples. A testing sample is firstly assigned to one of the regions, and the linear regression formula for that particular region is used to estimate its output value.

4.2.2 An Illustrative Example

In order to better illustrate the training of the proposed regression method, a simulation model is taken from literature. In brief, the illustrative example [217] describes the operation of a continuous stirred tank reactor, where a chain reaction of $A \rightarrow B \rightarrow C$ takes place. An inlet stream containing both reactant A and B enters the reactor and the desirable output is component B. There are 4 independent input variables to the simulation model, including temperature of the reactor (T), volume of the reactor (V), concentration of A and B in the inlet stream (CA^{in} and CB^{in}). The output to be predicted is the production rate of B (P). The process and associated variables are described in Figure 4.3.

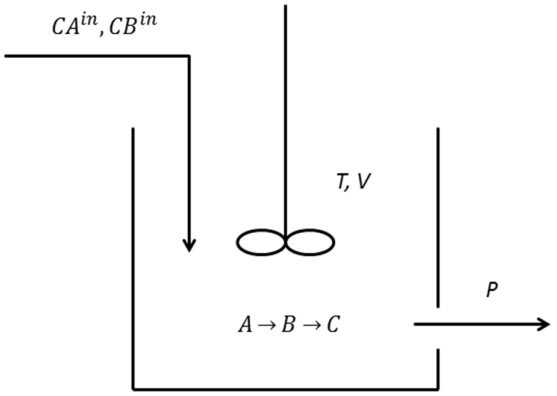

FIGURE 4.3: Illustrative example of a continuous stirred tank reactor. The inlet stream contains both reactant A and B. The chain reaction $A \rightarrow B \rightarrow C$ takes place within the tank with the reaction kinetics known. The desired output is component B.

With latin hypercube sampling technique [218] employed to specify a set of data points, we run the simulation model and collect 300 samples. The goal of the regression analysis is to approximate the functional relationship between output variable P and input variables including T, V, CA^{in} and CB^{in} using piece-wise linear functions. The step-wise description of the training procedure is presented in Table 4.1 below.

Initially, a linear regression function is fitted to the entire dataset without feature segmentation, which gives a sum of absolute deviation of 1677.78. The second iteration of the method solves 4 independent OPLRA models allowing 2 regions each, respectively specifying T, V, CA^{in} and CB^{in} as partitioning feature. The

two-region piece-wise linear functions constructed while partitioning on T appears to yield lower training errors (i.e. 1030.63) than the other 3, and therefore is taken as the solution for iteration 2. This represents a significant improvement (i.e. $\frac{1677.78-1030.63}{1677.78} = 38.57\%$) from the initial global linear regression function. From iteration 3, the partitioning feature is fixed as T while one more region is allocated for each increased iteration. Iteration 3 and 4 respectively lowers the training error to 876.66 and 807.12. The iterative procedure terminates when the β criterion is not satisfied, e.g. if $\beta = 20\%$, then the iterative procedure terminates at the third iteration and the final regression function has 2 regions; if $\beta = 10\%$, then the final regression function has 3 regions.

TABLE 4.1: Piecewise regression functions built at each step of training procedure

Iteration	Number of regions	Partition feature	Training error	Training error improvement	Functional relationship
1	1	NoneE	1677.78		$P = 1.0240T + 0.0054CA^{in} + 0.0125CB^{in} + 0.4340V - 333.54$
2	2	T	1030.63	38.57%	$P = \begin{cases} 0.7413T + 0.0040CA^{in} + 0.0102CB^{in} + 0.3406V - 238.74, & T \leq 213.21 \\ 1.7156T + 0.0111CA^{in} + 0.0315CB^{in} + 0.7574V - 592.63, & T > 213.21 \end{cases}$
	2	V	1143.49		$P = \begin{cases} 0.5952T + 0.0033CA^{in} + 0.0056CB^{in} + 0.4533V - 194.26, & V \leq 42.38 \\ 1.4781T + 0.0083CA^{in} + 0.0195CB^{in} + 0.4773V - 48.70, & V > 42.38 \end{cases}$
	2	CA^{in}	1485.65		$P = \begin{cases} 0.8930T + 0.0057CA^{in} + 0.0152CB^{in} + 0.4161V - 293.45, & CA^{in} \leq 3528.43 \\ 1.4857T + 0.0073CA^{in} + 0.0070CB^{in} + 0.5929V - 489.45, & CA^{in} > 3528.43 \end{cases}$
	2	CB^{in}	1627.73		$P = \begin{cases} 1.0242T + 0.0056CA^{in} + 0.0118CB^{in} + 0.4241V - 333.49, & CB^{in} \leq 458.21 \\ 1.1105T + 0.0050CA^{in} - 0.1405CB^{in} + 0.5813V - 291.00, & CB^{in} > 458.21 \end{cases}$
3	3	T	876.66	14.94%	$P = \begin{cases} 0.5815T + 0.0030CA^{in} + 0.0097CB^{in} + 0.2654V - 184.45, & T \leq 303.25 \\ 1.1353T + 0.0062CA^{in} + 0.0176CB^{in} + 0.4579V - 373.68, & 303.25 < T \leq 316.62 \\ 1.8764T + 0.0119CA^{in} + 0.0394CB^{in} + 0.8617V - 654.41, & T > 316.62 \end{cases}$
4	4	T	807.12	7.93%	$P = \begin{cases} 0.5815T + 0.0030CA^{in} + 0.0097CB^{in} + 0.2654V - 184.45, & T \leq 303.25 \\ 1.2648T + 0.0054CA^{in} + 0.0148CB^{in} + 0.4510V - 409.61, & 303.32 < T \leq 312.21 \\ 1.4872T + 0.0084CA^{in} + 0.0202CB^{in} + 0.6667V - 503.10, & 312.21 < T \leq 320.77 \\ 1.9930T + 0.0128CA^{in} + 0.0360CB^{in} + 0.8871V - 695.65, & T > 320.77 \end{cases}$
...					

Overall, the key features of our proposed piecewise linear regression method are summarised here: 1) our method identifies one key partitioning feature and separates samples into multiple complementary regions on it, 2) each region has the flexibility of being fitted by its own linear regression function, with all input features allowed to have different regression coefficients across different regions, 3) there is only one tuning parameter β, 4) compared with algorithms like kernel-based SVR and MLP, the constructed regression function is easy to understand,

as it exhibits linear relationships for different regions. In the next section, a number of real world regression problems are employed to benchmark the predictive performance of our proposed model.

4.3 Results and Discussion

A total number of 6 real world datasets have been downloaded from UCI machine learning repository (http://archive.ics.uci.edu/ml/) [219] to test the prediction performance of our proposed method. The first regression problem Yacht Hydrodynamics predicts the hydrodynamic performance of sailing yachts from 7 features describing the hull dimensions and velocity of the boat for 308 samples. Energy Efficiency [220] collects data corresponding to 768 building shapes, described by 8 features including wall area, root area and so on. The aims are to establish the relationships between either heating or cooling requirements and the 8 parameters of the building. The third example, Concrete Strength [221], looks into the relationship between compressive strength of concrete and 8 input variables, including water concentration and age, with 1030 samples of different concretes. Airfoil dataset concerns how the different airfoil blade desings, wind speed and angles of attack affect the sound pressure level. The last case study, White Wine Quality [173], aims to predict experts' preference of white wine taste with 11 physicochemical features of the wines. Around 4900 white wine samples have been obtained for analysis.

For each of the 6 benchmark datasets, a 5-fold cross validation, is performed to estimate the predictive accuracy of the proposed method. Given a dataset, 5-fold cross validation randomly splits the samples into 5 subsets of equal size. Each subset is in turn held out once while the other 4 subsets of samples are merged and used as the training samples to derive the regression function. The holdout set is then used to validate the predictive accuracy of the constructed regression function. We conduct 10 rounds of 5-fold cross validation by performing different random sample splits, and the mean absolute prediction errors (MAE) are averaged over 50 testing sets as the final error. The smaller the error, the better the prediction accuracy of a regression method.

For comparison purposes, a number of state-of-the-art regression methods have

been implemented, including linear regression, MLP, kriging, SVR, KNN, MARS, PaceRegression and ALAMO. Among all, Linear regression, MLP, kriging, SVR, KNN and PaceRegression are implemented in WEKA machine learning software [109]. Linear regression is performed with backward feature selection using M5' algorithm, which is to ensure generalisation of the constructed model to unseen samples when the number of features is large. For KNN, the number of nearest neighbours is selected as 5. With regards to other methods, their default settings have been retained. In WEKA, default setting for MLP includes: learning rate=0.3, momentum=0.2, one single hidden layer with number of neurons equal to the number of features divided by 2, and training time=500. For SVR, complexity coefficient is set as 1 and polynomial kernel with exponent of 1 is used. The MATLAB toolbox called ARESlab [222] is implemented for MARS, with second order interaction between features permitted. ALAMO is reproduced using the General Algebraic Modeling System (GAMS) [70], and basis function forms including polynomial of degrees up to 3, pair-wise multinomial terms of equal exponents up to 3, exponential and logarithmic forms are provided for each dataset. Our proposed method is also implemented in GAMS. Both ALAMO and our proposed model are solved using Cplex MILP solver, with optimality gap set as 0. Computational resource limit is set as 200 seconds for each solving of OPLRA model in our proposed method, and each iterative procedure in ALAMO.

4.3.1 Sensitivity Analysis for β

In this subsection, a sensitivity analysis is performed for the parameter β, which serves as a terminating criterion of the iterative training procedure for our proposed method. Taking value between *0* and *1*, β defines the minimum percentage training error reduction that must be achieved to justify the allocation of an extra region. A range of values have been tested, including: *0.2, 0.15, 0.10, 0.05, 0.03* and *0.01*. The results of the sensitivity analysis are provided in Figure 4.4.

Figure 4.4 describes how mean absolute error changes with β. The numbers attached to the points in each plot are the average numbers of final regions over 50 training runs, which always go up as β decreases. For Yacht Hydrodynamics example, setting $\beta = 0.20$ results in just more than 4 final regions. Decrease the β value to *0.15* increases slightly the prediction error with marginally higher number of regions. Further decrease β to *0.10* leads to lowest mean prediction error of

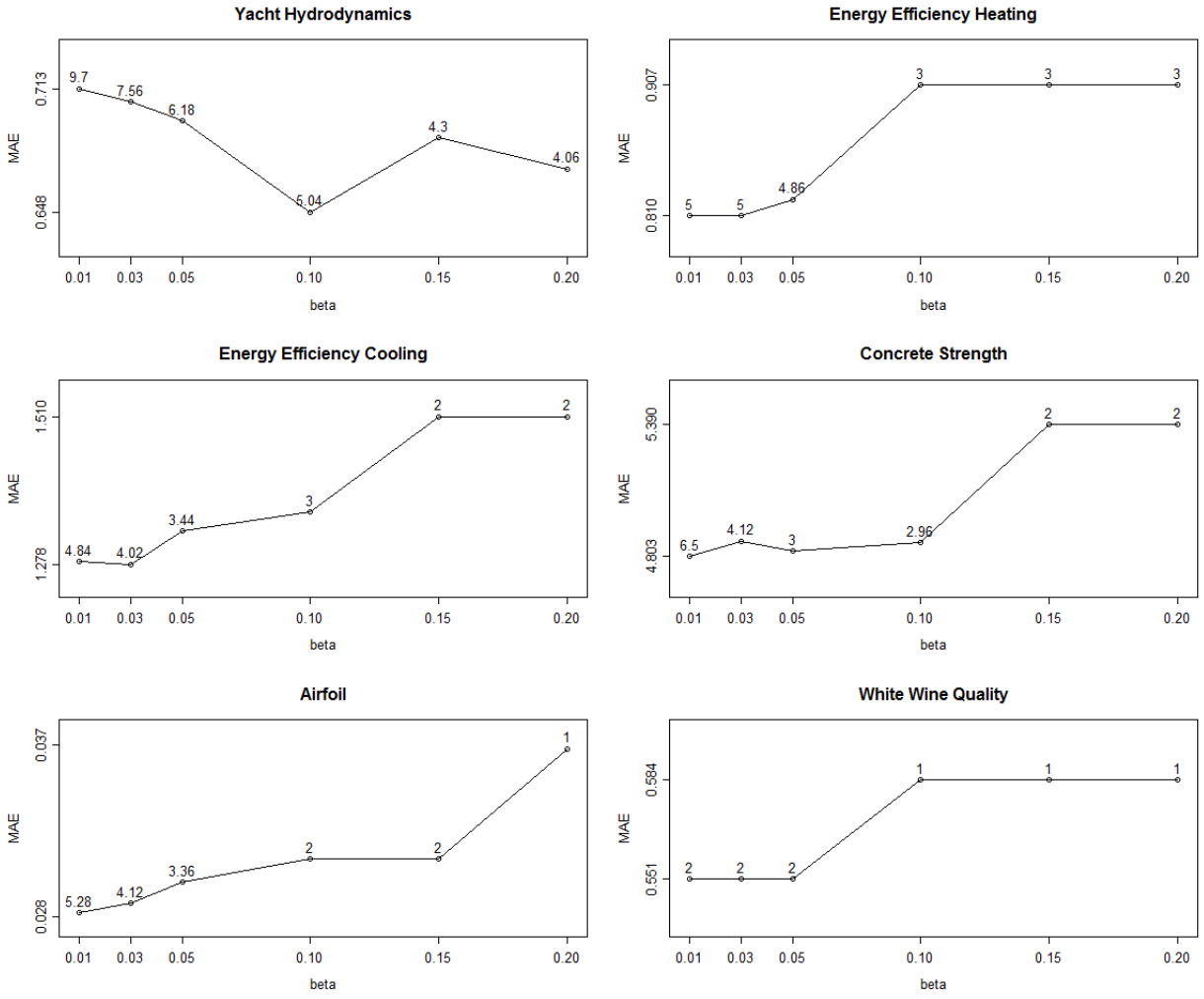

FIGURE 4.4: Sensitivity analysis of β for the proposed piece-wise linear regression method. Each line describes how mean absolute prediction error varies with different values of β. The numbers above points in each plot correspond to the average numbers of final regions over 50 training runs.

0.648 with an average of 5 regions, before excessively low values of β over-fits the unseen testing samples by yielding much increased prediction error. For Energy Efficiency Heating case study, when $\beta = 0.10, 0.15$ and 0.20 our proposed regression method constructs piece-wise regression functions of an average of 3 regions, yielding MAE of 0.907. Smaller values of β leads to about 5 regions, which are shown to predict the testing samples with higher accuracy (MAE around 0.810). In terms of Energy Efficiency Cooling and Concrete Strength examples, similar phenomenon can be observed that when β takes excessively high values (i.e. 0.20, 0.15), the proposed method terminates prematurely with only 2 regions and relatively high MAE. More regions are allowed by lowering β, which gives higher prediction accuracies. On Airfoil case study, the proposed method outputs global

multiple linear regression functions without data partitions when $\beta = 0.20$. As β decreases, more regions are permitted, which predict unseen samples with better accuracy. On the last example of White Wine Quality, 2-region piece-wise regression functions achieved with $\beta = 0.01, 0.03, 0.05$ outperforms global multiple linear regressions for higher values of β.

It can be seen from Figure 4.4 that the range of values between *0.01* and *0.05* generally lead to smaller prediction error than higher values of β. For all datasets except Yacht Hydrodynamics, prediction errors of $\beta = 0.01, 0.03$ and *0.05* are evidently smaller than that of $\beta = 0.10, 0.15$ and *0.20*. Within the range between *0.01* and *0.05*, there is no clear optimal value for β as different values have different effects on the accuracy. The consistently small MAE, while β is between *0.01* and *0.05*, show that our proposed regression method is robust with respect to the only user tuning parameter β. Finally, when comparing with other competing methods in literature, β is set to *0.03* which gives consistently desirable prediction accuracy across a wide range of problems.

4.3.2 Prediction Performance Comparison

After identifying a value (i.e. *0.03*) for the only tuning parameter β in our proposed regression method, we now compare the accuracy of the proposed method against some popular regression algorithms with the same set of 6 examples. The results of the comparison are available in Table 4.2 below.

Table 4.2: Comparative testing of different regression methods on benchmark datasets

	Yacht Hydrodynamics	Energy Efficiency Heating	Energy Efficiency Cooling	Concrete Strength	Airfoil	White Wine Quality
linear regression	7.270	2.089	2.266	8.311	0.037	0.586
MLP	0.809	0.993	1.924	6.229	0.035	0.623
Kriging	4.324	1.788	2.044	6.224	0.030	0.576
SVR	6.445	2.036	2.191	8.212	0.037	0.585
KNN	5.299	1.937	2.148	7.068	**0.026**	**0.537**
MARS	1.011	**0.796**	1.324	4.871	0.035	0.570
PaceRegression	7.233	2.089	2.261	8.298	0.037	0.586
ALAMO	0.787	2.722	2.765	8.044	0.032	0.639
Proposed	**0.706**	0.810	**1.278**	**4.870**	0.029	0.551

*The lowest MAE for each dataset is highlighted in bold.

On Hydrodynamics problem, the proposed method in this work provides an MAE

of 0.706, which is lower than any other competing algorithm. ALAMO, MLP and MARS follow closely with MAE of 0.787, 0.809 and 1.011, respectively. Mean error rates of the rest of the methods are significantly higher and between 3 and 8. On Energy Efficiency Heating, MARS emerges as the most accurate algorithm with an mean absolute error of 0.796, which is closely matched by the proposed method and MLP. Mean prediction errors of the other approaches are almost all twice as large as that of the MARS. In terms of Energy Efficiency Cooling dataset, the proposed method, MARS and MLP are the top 3 performers with MAE between 1.278 and 1.924. On Concrete Strength, the proposed approach and MARS, with an MAE of 4.870 and 4.871, again emerge as the leading methods from Kriging, MLP and the others. When it comes to Airfoil example, all the competing algorithms achieve similar prediction accuracies, with KNN topping the league with an MAE of 0.026. The proposed approach in this work is a merely 0.003 far behind, with kriging a further 0.001 behind. A mere difference of 0.011 separates the 10 methods. Lastly, on the White Wine Quality example, the proposed approach is ranked as the second best method after KNN.

Overall, for 3 out of the 6 datasets, including Yacht Hydrodynamics, Energy Efficiency Cooling and Concrete Strength, the proposed piece-wise regression method achieves the lowest prediction errors. For the other 3 tested examples, including Energy Efficiency Heating, Airfoil and White Wine Quality, the proposed method still performs competitively as being second on all of them.

As there does not exist a single regression method which can always outperform others on all datasets, a desirable regression algorithm should demonstrate consistently competitive prediction accuracy. In order to more comprehensively evaluate the relative competitiveness of all the implemented approaches, we employ the following scoring strategy: for each problem, the regression methods are ranked in descending order according to their mean prediction error. The best regression method corresponding to the lowest prediction error is awarded the maximum score of 9, the second best regression method corresponding to the second lowest prediction error is assigned a score of 8 and so on. The scores of each regression approach are averaged over the 6 datasets, which represent the overall performance of the method. The higher the score, the better the relative performance of a method. The scores of the different regression approaches used in this work are presented in Figure 4.5 below.

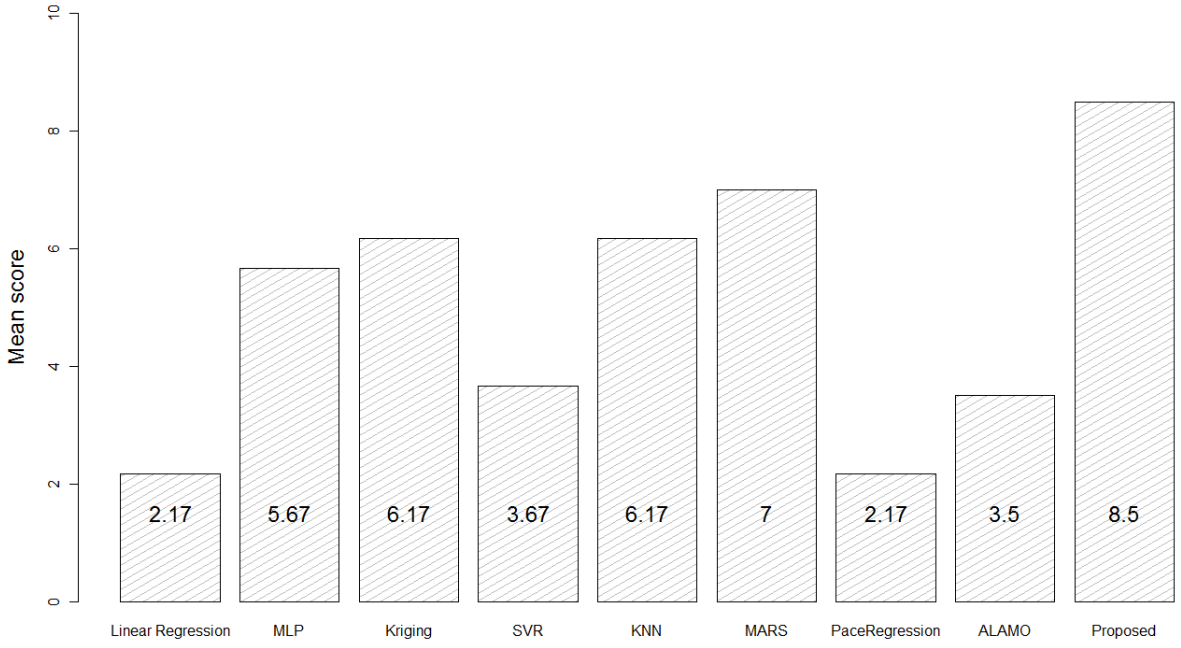

FIGURE 4.5: Scoring of regression methods. A scoring strategy is employed to evaluate the overall prediction performance of the implemented methods.

According to Figure 4.5, the proposed method is shown to be the most accurate and robust regression algorithm among all, achieving a score of 8.5 out of a possible 9. MARS is second with scores of 7, followed by Kriging, KNN and MLP in descending order. The advantages of the proposed regression method is quite obvious compared with other implemented methods.

Lastly, we apply the proposed piece-wise linear regression model to all the available samples in each dataset and take a look at the number of regions and the key partitioning feature determined. The results are summarised in Table 4.3. It is clear that the proposed segmented regression method provides good interpretability as the number of regions are small (usually between 2 to 4 and at most 5). The partitioning features may release important insights into the underlying system as the output variables change more dramatically across different ranges alone this feature.

TABLE 4.3: Number of regions and partition feature by our proposed method

Dataset	Number of regions	Partitioning feature
Hydrodynamics	5	Froude number
Energy Heating	3	Wall Area
Energy Cooling	3	Wall Area
Concrete	3	Age
Airfoil	4	Frequency
White Wine	2	Volatile acidity

4.3.3 Piece-wise Linear Regression May Serve as A Surrogate Model

Besides desirable prediction accuracy, the simplicity and the explicit algebraic forms of the proposed piece-wise linear regression method make it possible for it to be further applied as a surrogate model. For example, in a typical chemical engineering modular flowsheet process, where different unit operations are sequentially connected to perform a task, global optimisation of design or operational variables poses great computational difficulty [175]. The reasons are that the individual unit operations can be highly complex and non-linear, usually making the optimisation of the entire flowsheet a difficult mixed integer non-linear programming problem, where quality solutions are hard to obtain. Under those circumstances, the proposed method in this chapter can serve as a surrogate model by replacing the computationally troublesome unit operations with much simpler piece-wise linear function, yet accurately mimics the behaviour of the original model.

To briefly demonstrate this applicability, the illustrative example discussed in section 4.2.2 is reviewed. In the illustrative example, the underlying simulation model consists of several non-linear equations that describe the relationship between production rate of reactant B (P) and 4 independent input variables, i.e. concentration of both reactant A and B (CA^{in}, CB^{in}, respectively), temperature of reactor (T) and volume of reactor (V). Assuming 4 intervals ($R = 4$), our proposed segmented regression method conveniently approximates the original simulation model with the following block of equations. The first of those

equations are written below:

$$\begin{aligned} 0 < & T_{r1} \leq 303.25 E_{r1} \\ 303.25 E_{r2} < & T_{r2} \leq 312.21 E_{r2} \\ 312.21 E_{r3} < & T_{r3} \leq 320.77 E_{r3} \\ 320.77 E_{r4} < & T_{r4} \leq U E_{r4} \end{aligned} \tag{4.9}$$

where r denotes the interval and E_r is binary variable that are equal to 1 if temperature of reaction is in region r, defined by the two consecutive break-points; 0 otherwise. T_r is equal to the reaction temperature T if T falls into region r, and is equal to 0 otherwise. U is an arbitrarily large number used to bound T_{r4}. Temperature of the reaction can only fall into one of the intervals:

$$\sum_{r=1}^{4} E_r = 1 \tag{4.10}$$

Each different temperature interval has its own multiple linear function:

$$\begin{aligned} P_{r1} &= 0.5815 T_{r1} + 0.0030 CA_{r1}^{in} + 0.0097 CB_{r1}^{in} + 0.2654 V_{r1} - 184.45 E_{r1} \\ P_{r2} &= 1.2648 T_{r2} + 0.0054 CA_{r2}^{in} + 0.0148 CB_{r2}^{in} + 0.4510 V_{r1} - 409.61 E_{r2} \\ P_{r3} &= 1.4872 T_{r3} + 0.0084 CA_{r3}^{in} + 0.0202 CB_{r3}^{in} + 0.6667 V_{r1} - 503.10 E_{r3} \\ P_{r4} &= 1.9930 T_{r4} + 0.0128 CA_{r4}^{in} + 0.0360 CB_{r4}^{in} + 0.8871 V_{r1} - 695.65 E_{r4} \end{aligned} \tag{4.11}$$

where CA_r^{in}, CB_r^{in} and V_r are all intermediate variables that equal to CA^{in}, CB^{in} and V, respectively, if temperature T is in region r ($E_r = 1$), and equal to 0 otherwise:

$$0 \leq CA_r^{in} \leq U E_r \qquad \forall r \tag{4.12}$$

$$0 \leq CB_r^{in} \leq U E_r \qquad \forall r \tag{4.13}$$

$$0 \leq V_r \leq U E_r \qquad \forall r \tag{4.14}$$

Finally, the intermediate variables need to be linked to the original variables:

$$P = \sum_{r=1}^{4} P_r \tag{4.15}$$

$$CA^{in} = \sum_{r=1}^{4} CA_r^{in} \tag{4.16}$$

$$CB^{in} = \sum_{r=1}^{4} CB_r^{in} \tag{4.17}$$

$$T = \sum_{r=1}^{4} T_r \tag{4.18}$$

$$V = \sum_{r=1}^{4} V_r \tag{4.19}$$

The block of equations from (4.9) to (4.19) are linear and involves merely 4 binary variables (E_r), and thus is computationally cheap to evaluate. The feature of the piece-wise linear regression functions to be used as surrogate models will be investigated with real world case studies as future work.

4.4 Concluding Remarks

In this chapter, we have proposed a new method for multivariate data regression problem. The method separates samples into multiple regions by segmenting a single key partition feature, while simultaneously fitting one linear regression function per region. Assuming both partition feature and number of break-points are known, a novel MILP model, which optimally determines the locations of break-points and regression coefficients for each region corresponding to least absolute deviation, has been presented. A heuristic procedure has been used to find the key partition feature and the number of break-points.

Six benchmark regression datasets, from various application domains and of different sizes (number of samples from a few hundreds to up to 5000), have been used to demonstrate the applicability and efficiency of the proposed method. Comparing against a number of popular regression methods in literature, including kriging, MARS, SVR and MLP, it is shown that our proposed method achieves consistently high prediction accuracy as leading to the lowest prediction errors for 3 out of 6 datasets, and second lowest errors for the other 3 datasets. The results confirm our proposed method as a reliable alternative to traditional regression analysis methods in literature.

Besides the acceptable prediction accuracy, our proposed method is much more interpretable than most of the machine learning algorithms, for example kernel-based support vector machine, neural network, statistical methods of kriging, as it approximates the input-output relationship as a piece-wise linear algebraic formula. Therefore the proposed method in this work can also be used as surrogate model and help solve complex chemical engineering-based flowsheet optimisation problems, where the behaviour of each unit can be complicated and highly non-linear, making the whole process extremely difficult to optimise directly. In this case, the piece-wise linear regression can approximate the complex relationship between design variables with simpler piece-wise algebraic functions for each unit separately. Global optimisation can then be carried on the simplified process model.

Chapter 5

A Novel Regression Tree Model

In the last chapter, a piece-wise linear regression model (OPLRA) has been proposed, which partitions a key feature into several intervals and fits one unique multivariate linear regression model for each interval. One of the shortcomings of the method is that only one feature is segmented, limiting its application to model more complex relationships. In this chapter, a novel solution procedure is proposed so that segmentation of multiple features is made possible. The resulting input-output relationship becomes a regression tree, i.e. an variant of decision tree for classification.

5.1 Introduction and Literature Review

Since the novel solution procedure introduced later in this chapter learns a tree-like regression model, different regression tree-based learning algorithms are firstly reviewed in this section, including classification and regression tree (CART), M5', Cubist, smoothed and unsmoothed piecewise-polynomial regression trees (SUPPORT) and generalised, unbiased interaction detection and estimation (GUIDE).

5.1.1 CART

CART [201] is probably the most well known regression tree learning algorithm in literature. Given a set of samples, CART identifies one input variable and one break-point, and partitions the samples into two child nodes. Starting from

the entire set of available training samples (root node), recursive binary partition is performed for each child node until no further split can enhance the training performance or another terminating criteria is satisfied. At each node, best split is identified by exhaustive search, i.e. all potential splits on each input variable and each break-point are tested, and the split corresponding to the minimum deviations, obtained by respectively predicting two child nodes of samples with their mean output variables, is selected. After the tree growing procedure, typically an excessively large tree is constructed, resulting in lack of model generalisation. A procedure of pruning is employed to remove one by one the splits contributing least to training accuracy. The tree is pruned from the maximal-sized tree all the way back to the root node, resulting in a sequence of candidate trees. The optimal candidate tree can be selected using external validation data or inner-cross validation [223, 224]. Given an enquiry sample, it is firstly assigned into one of the terminal leaves (non-splitting leaf nodes) and then predicted with the mean output value of the samples belonging to the leaf node. Despite the simplicity and good interpretation, the simple rule of predicting with mean output values at the terminal leaves often means prediction performance is compromised [225].

5.1.2 M5' and Cubist

M5' [226, 227] is considered an improved version of CART. The tree growing process is the same as that of the CART, while several modifications have been introduced in tree pruning process. After the full size tree is produced, a multiple linear regression model is fitted for each node. An empirical metric is proposed that estimates model generalisation error as a function of training error, the numbers of training samples and model parameters. The constructed linear regression function per node is then simplified by removing insignificant input variables using a greedy algorithm in order to achieve locally maximal model generalisation error metric. Starting from the bottom of the tree, pruning is tested for each non-terminal node. If the parent node offers higher model generalisation than the sum of its two child nodes, then the child nodes are pruned away.

Prediction procedure used by M5' also differs substantially from CART. Given a new sample, it is assigned to one of the terminal leaves. All nodes along the path from the root node to the terminal leave make their own predictions, which are aggregated using another empirical formula. The effect of this is to smooth the

predictions when the nodes along the path predict the new samples very differently. The modifications of fitting linear regression functions to each node, simplifying linear regression functions, pruning trees and smoothing predictions make M5' a much more powerful tool than the conventional CART.

Cubist [228] is rule-based regression model, which further extends the work of M5'. Cubist starts with building a M5' tree. Each path from the root node to a terminal leaf defines an unique rule. Those rules are then pruned and/or combined, which leads to a new set of simplified and possibly overlapping rules, i.e. the rules are not mutually exclusive and therefore a new sample may satisfy more than one rule. In this case the predicted values from all rules are averaged to yield the final prediction. The classification version of Cubist is called C5.0 [229] (https://www.rulequest.com/see5-info.html), which can express classification boundary as a collection of if-then decision rules, as Cubist does. C5.0 has been designed to have better scalability, memory efficiency and smaller tree size than its predecessor decision tree classifiers.

5.1.3 SUPPORT and GUIDE

SUPPORT [230] is another decision learning algorithm for regression analysis. Given a node, SUPPORT fits a multiple linear regression function and computes the residual of each sample. The samples with positive deviations and negative deviations are respectively assigned into two classes. For each input variable, SUPPORT compares the distribution of the two classes of samples along this input variable by applying two-sample t test. The input variable corresponding to the lowest P value, i.e. the most significant difference in the distributions of the two classes, is selected as splitting feature and the average of the two class means on this splitting variable is taken as break-point.

GUIDE [231] adopts similar philosophy as the SUPPORT. Given a node, the same step of fitting samples with a linear regression model and separating samples into two classes based on the sign of deviations is employed. For each input variable, its numeric values are binned into a number of intervals before a chi-square test is used to determine its level of significance. The most significant input variable is used for split. In terms of break-point determination, either a greedy search or

median of the two class mean on this splitting variable can be used. Other invariants of decision tree-based regression learning methods also exist in literature, including but not limited to: [20, 232–234].

Decision and regression trees can also be constructed using genetic programming. Evolutionary learning of globally optimal classification and regression trees (evtree) [235] employs evolutionary algorithms to derive trees while considering not only the next split but also potential splits further down the tree, attempting to identify a more optimised tree structure. Some other existing work in literature that induct tree models using genetic programming techniques include but are not limited to [236], [237] and [238].

The traditional means of node splitting are dominated by either exhaustively searching the candidate split corresponding to the maximum variance reduction by predicting using mean output values in either child node [201, 226, 227], or more sophisticated fashion of examining distribution of sample deviation from fitting one linear regression function to all the samples in the parent node [230, 231]. However, it is noticed that for those algorithms where terminal leaf nodes are fitted with linear regression functions [226–228], the determinations of splitting feature, break-point and regression coefficients are done sequentially, i.e. the splitting feature and break-point are estimated during tree growing procedure while regression coefficients for each child node are computed at pruning step.

An alternative node splitting strategy is to *simultaneously* optimise the splitting feature, the position of break-point and the linear regression coefficients for each parent node. In this case, the quality of a split can be directly calculated as the residual sum of all samples in either subset. A straightforward exhaustive search algorithm for this problem can be: for each input variable and each possible break-point, samples are separated into two subsets and one multiple linear regression function is fitted for each subset. After examining all possible splits, the optimal split is chosen as the one corresponding to the minimum sum of residual. The problem with this approach is, however, that as the numbers of samples and input variables grow, the quantity of multiple linear regression functions need to be evaluated increases exponentially, requiring infeasible computational time. For example, given a regression problem of 500 samples and 10 input variables and

assuming for each input variable, each sample takes an unique value, then it requires construction of 499*10*2 multiple linear regression functions in order to find the optimal binary split for the root node. The amount of computational time required to grow a maximal-sized tree will prove to be unaffordable.

In this work, we adopt the OPLRA model from last chapter, which can solve the problem of binary splitting to *global* optimality in affordable computational time. A novel recursive splitting procedure is introduced to grow a regression tree. Recursive splitting terminates when the amount of reduction in training error achieved by node splitting is below an user-specific value, which is also the only tuning parameter in the proposed method. Since the size of tree is manually controlled via the tuning parameter, a pruning procedure is not implemented. The benchmark regression datasets introduced in the last chapter have been used to demonstrate the efficiency of the proposed regression-tree method.

5.2 A Novel Regression Tree Model

In this section, detailed descriptions of a novel recursive partitioning method used to construct a regression tree model and the procedure for predicting a new sample after building a tree are provided.

5.2.1 A Novel Recursive Partitioning Tree Growing Method

Similar to almost all other decision tree learning algorithms, recursive partitioning is used to grow the tree from root node until a split of node cannot yield sufficient reduction in deviation. The pseudocode for building a tree is given below.

Given training samples, the first step of our proposed tree growing strategy is to fit a multiple linear regression function to the entire set of training samples minimising absolute deviation, which is noted as $ERROR_{root}$. The absolute deviation of root node multiplying a scaling parameter β, taking value between *0* and *1*, is specified as the condition for node splitting, i.e. a node is split into two child nodes only if the optimal split of the node results in reduction in absolute deviation greater than $\beta * ERROR_{root}$. Then starting from the root node, each feature m is specified in turn as splitting feature m^* once, while solving model OPLRA

Tree growing algorithm

1. Fit a multiple linear regression model to root node containing all training samples minimising absolute deviation, recorded as $ERROR_{root}$.
2. Starts from the root node.
3. For each input feature m, specify it as splitting feature ($m = m^*$) and solve model OPLRA while allowing two regions ($R = 2$). The deviation is noted as $ERROR_m$.
4. Find the best split corresponding to the minimum absolute deviation, noted as $ERROR_{split} = \min_m ERROR_m$.
5. If $ERROR_{parent} - ERROR_{split} \geq \beta * ERROR_{root}$, the current node is split. $ERROR_{parent}$ is simply the absolute deviation of multiple linear regression on the parent node and β is user-specific parameter.
6. Apply step 3-5 to each child node in turn.

minimising the sum of absolute deviations of two child nodes. The best split of the current node is identified as the one corresponding to minimum absolute error. If the best split brings down absolute deviation from its parent node by more than $\beta * ERROR_{root}$, then the split takes place; otherwise the current node is finalised as terminal leaf node.

A common problem that regression tree methods face is the non-smoothness of the derived models. In other words, as the decision space is partitioned into disjoint regions and each region is fitted with a constant (CART) or a multivariate function (M5'), the predicted output values from both child nodes usually are sharply different near the break-point. This leads to non-smoothness of the model and decreased prediction accuracy. In terms of the proposed ORTREE, model non-smoothness is moderate. When the density of the samples is sufficiently large so that there is no void in the decision space, multivariate functions from two child nodes nearly meet each other at the break-point, giving to a reasonably smooth model (data not shown).

5.2.2 Predicting New Samples

After building a regression tree, predicting the level of output variable for a new sample is intuitive. A new sample is assigned to one of the terminal leaf nodes, before predicting it using the multiple linear regression function derived for this terminal node in the training procedure. The predicted output value is also bounded

by the minimum and the maximum output values of the training samples belonging to this particular terminal node. In other words, if the predicted value is greater than the maximum output value (or smaller than the minimum output value) among all the training samples in this terminal node, the prediction is modified to equal to the maximum output value (or the minimum output value).

The proposed regression tree method, referred to as ORTREE (Optimal Regression TREE), is applied to the real world regression problems introduced in the last chapter to demonstrate its applicability and efficiency.

5.3 Results and Discussion

In this section, we aim to evaluate the behaviour of the proposed ORTREE using all the 6 real world benchmark datasets introduced in the last chapter, including Yacht Hydrodynamics, Energy Efficiency Heating, Energy Efficiency Cooling, Concrete Strength, Airfoil and White Wine Quality. A comprehensive sensitivity analysis for the tuning parameter β is firstly conducted in order to identify a robust value that gives consistently good prediction accuracy. After that, prediction accuracy comparison is performed to evaluate ORTREE against certain decision tree learning algorithms and some other regression methodologies. Lastly, the interpretability of the constructed tree models are evaluated by examining the number of terminal leaf nodes.

To assess the relative competitiveness of the proposed ORTREE in terms of prediction accuracy, some decision tree learning algorithms and regression methods based on various other methodologies have been implemented for comparison. More specifically, we compare the proposed ORTREE to CART, M5', Cubist, linear regression, MLP, Kriging, SVR, KNN, MARS, PaceRegression, ALAMO and segmented regression, which is proposed in the last chapter. CART and Cubist are implemented in R [213] using the package 'rpart' [239] and 'Cubist' [228], respectively. M5', Linear regression, SVR, MLP, kriging and KNN are implemented in WEKA machine learning software [109]. For KNN, the number of nearest neighbours is selected as 5, while for other methods their default settings have been retained. The MATLAB toolbox called ARESlab [222] is used for MARS. ALAMO is reproduced using the General Algebraic Modeling System (GAMS)

[70], and basis function forms including polynomial of degrees up to 3, pair-wise multinomial terms of equal exponents up to 3, exponential and logarithmic forms are provided for each dataset. Segmented regression and the proposed ORTREE are also implemented in GAMS. ALAMO, segmented regression and ORTREE are solved using Cplex MILP solver, with optimality gap set as 0.

5.3.1 Sensitivity analysis for β

In this section, we first perform a comprehensive sensitivity analysis on the single tuning parameter β in the proposed ORTREE. Recall in the tree growing procedure, β controls termination of recursive node splitting. A node is split into two child nodes if the optimal split leads to reduction in absolute training deviation of more than a threshold value, defined as the amount of absolute training deviation of a multiple linear regression analysis on the entire set of training samples ($ERROR_{root}$) multiplying the scaling parameter (β). The tree grows larger as β decreases. Identifying a suitable value for β is a non-trivial problem as an excessively high value would terminate the node splitting prematurely without adequately describing the data, while an excessively small value can over-fit the unseen samples by constructing a very large tree. In this work, we test a series of values, including *0.05, 0.025, 0.015, 0.01, 0.005, 0.0025* and *0.001*. The results of the sensitivity analysis are presented in Figure 5.1.

According to Figure 5.1, we can clearly observe a phenomenon that as β goes down from *0.05* to *0.01*, prediction error keeps declining. This improved prediction accuracy is attributed to the fact that decreased β allows the tree to grow larger, and thus better describing the pattern in the data. MAE appears to vary in different patterns in different datasets with lower values of β. For example, on Yacht Hydrodynamics, Concrete Strength and White Wine Quality, error rate monotonically goes up as β decreases, suggesting that the constructed trees are too large and over-fit the unseen samples. On the other 3 datasets, MAE drops initially with lower value of β between *0.01* and *0.0025*, before prediction becomes significantly worse at $\beta = 0.001$.

It is well known that in data mining, parameter fine tuning is required for a particular method to reach optimal performance for a specific dataset. Thus, it is our interest here to identify a value for β that corresponds to robust prediction

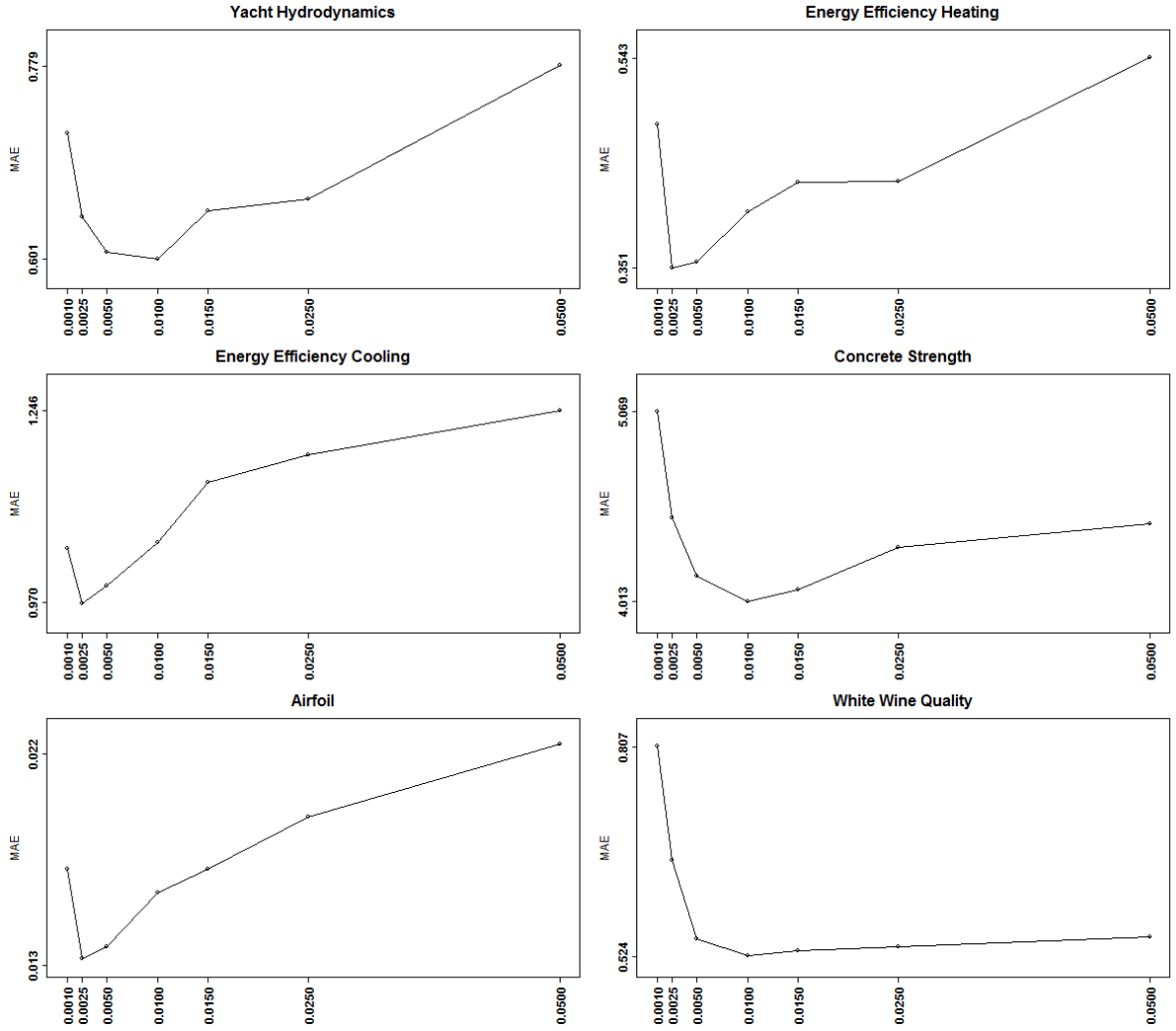

FIGURE 5.1: Sensitivity analysis of β for ORTREE. Each line describes how mean absolute prediction error varies with different values of β.

accuracy for a range of different tested benchmark examples. In this study, $\beta = 0.005$ appears to be a satisfactory value, as its prediction accuracies are the second best on 4 datasets and third best on the other 2 datasets, giving the most stable performance. MAE of higher and lower values of β are much less stable, e.g. when $\beta = 0.0025$, MAE are noticeably higher than the lowest MAE on Yacht Hydrodynamics, Concrete Strength and White Wine Quality, even though it yields the lowest MAE for the other 3 datasets. The same instability can be observed for $\beta = 0.01$. Overall, it is concluded that $\beta = 0.005$ is a suitable parameter setting for the proposed ORTREE.

5.3.2 Performance comparison across different regression methods

After identifying a value (i.e. *0.005*) for the only user-specific parameter, β, in the proposed ORTREE, we now compare the prediction performance of the ORTREE to a number of state-of-the-art regression methods. To ensure unbiased comparison, β is set to *0.005* thorough the comparative study. For each of the benchmark examples, we compare the MAE achieved by various competing methods, the results of which are summarised in Table 5.1.

TABLE 5.1: Prediction accuracy (MAE) comparison across different regression methods

	Yacht Hydrodynamics	Concrete Strength	Energy Efficiency Heating	Energy Efficiency Cooling	Airfoil	White Wine Quality
Tree-based regression methods						
ORTREE	0.608	**4.159**	0.356	0.995	**0.014**	0.547
CART	1.607	7.224	2.000	2.385	0.035	0.603
M5'	0.959	4.722	0.692	1.205	0.021	0.561
Cubist	**0.603**	4.289	**0.351**	**0.887**	0.017	0.546
Non-tree-based regression methods						
Linear regression	7.270	8.312	2.089	2.266	0.037	0.586
MLP	0.809	6.229	0.993	1.924	0.035	0.623
Kriging	4.324	6.224	1.788	2.044	0.030	0.576
SVR	6.445	8.212	2.036	2.191	0.037	0.585
KNN	5.299	7.068	1.937	2.148	0.026	**0.537**
MARS	1.011	4.872	0.796	1.324	0.035	0.570
PaceRegression	7.233	8.298	2.089	2.261	0.037	0.586
ALAMO	0.787	8.044	2.722	2.765	0.032	0.639
Segmented regression	0.706	4.870	0.810	1.278	0.029	0.551

Three major findings can be noticed from Table 5.1. Firstly, ORTREE always leads to smaller prediction errors than segmented regression. This is hardly a surprise as ORTREE allows segmenting multiple features and therefore is capable of modelling more complex local non-linearity, while segmented regression can only partitions a single feature. Secondly, 3 tree-based regression methods, including ORTREE, M5' and Cubist clearly outperform counterparts based on other methodologies, including the segmented regression proposed in the last chapter, MARS, kriging, KNN and so on. In addition, Cubist appears to predict slightly better than ORTREE, as it beats the proposed ORTREE in 4 out of 6 problems, while ORTREE is more accurate in the other 2 problems.

More specifically, on Yacht Hydrodynamics dataset, the best performing method is

Cubist with an MAE of 0.603, followed closed by ORTREE and segmented regression with MAE of 0.608 and 0.706, respectively. On Concrete Strength dataset, ORTREE and Cubist again lead others with MAE of 4.159 and 4.289, respectively. M5', segmented regression and MARS are ranked between third and fifth with MAE being lower than 4.9. Similar phenomenons are observed for Energy Efficiency Heating, Energy Efficiency Cooling and Airfoil, where ORTREE and Cubist occupy the top two spots and offer very similar accuracies. M5' are the third best method for the above 3 datasets but the margin to the top 2 is obvious. On the last dataset of White Wine Quality, Cubist and ORTREE are ranked the second and third methods after KNN.

The scoring strategy used in the previous chapter is repeated here to produce an overall ranking of various regression methods over 6 datasets. In the current case, the best method is awarded a score of 13, which is the total number of regression methods implemented in this comparative study and the method corresponding to the largest MAE is given a score of 1. The mean scores and ranking of each method are given in Figure 5.2. Figure 5.2 supports our previous statements that Cubist, ORTREE and M5' are the top 3 methods, and segmented regression is ranked fourth and is the best among non-tree-based methods.

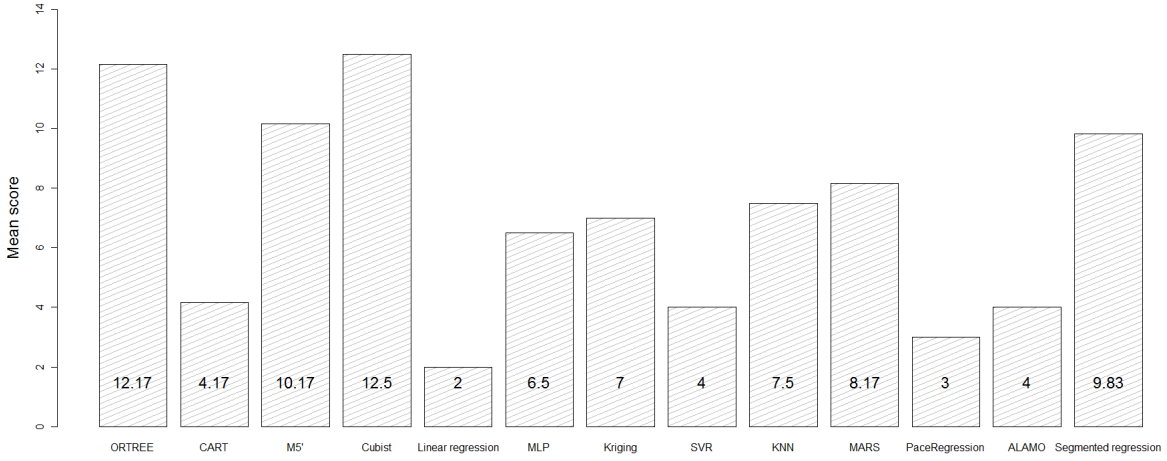

FIGURE 5.2: Scoring of tree-based and non-tree-based regression methods.

Overall, it is obvious from the comparison that the proposed ORTREE regression tree learning method has managed to closely match the prediction performance of the state-of-the-art methods in literature. ORTREE performs comparatively to Cubist and beat other methods by convincing margins.

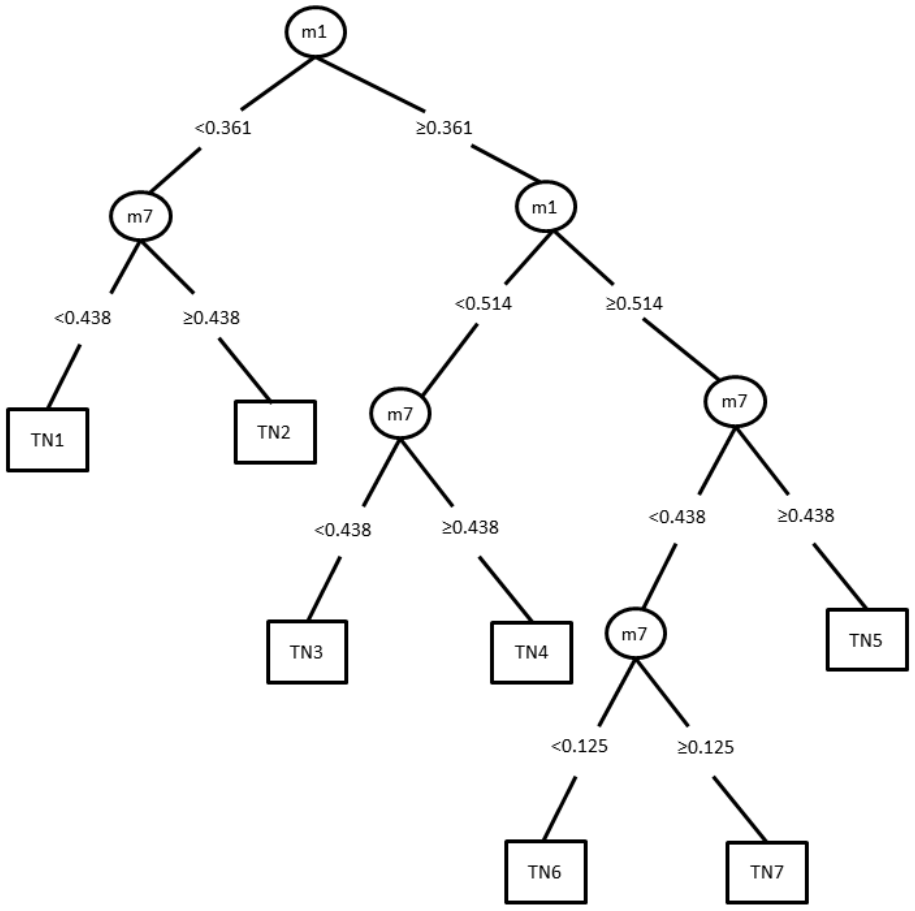

FIGURE 5.3: Constructed tree by CART on Energy Efficiency Heating example.

5.3.3 Comparison of actual constructed trees by different regression tree methods

Last section has demonstrated that the novel ORTREE regression tree learning method offers superior prediction capacity. Compared to certain regression methods whose output models cannot be interpreted, for example kernel-based SVR and MLP, tree learning algorithms are well-known for their easy interpretability. The sequence of the derived rules can be simply visualised as tree, making it easily understandable and possible to gain some insights into the underlying mechanism of the studied system. The interpretability of a constructed tree model decreases as the tree grows larger. In this section, attention is turned into comparing the number of terminal leaf nodes of the trees constructed by CART, M5' and ORTREE. Taking Energy Efficiency Heating as an example and using all the available samples as training set, the trees grown by CART, M5' and MPT are presented in Figures 5.3, 5.4 and 5.5, respectively.

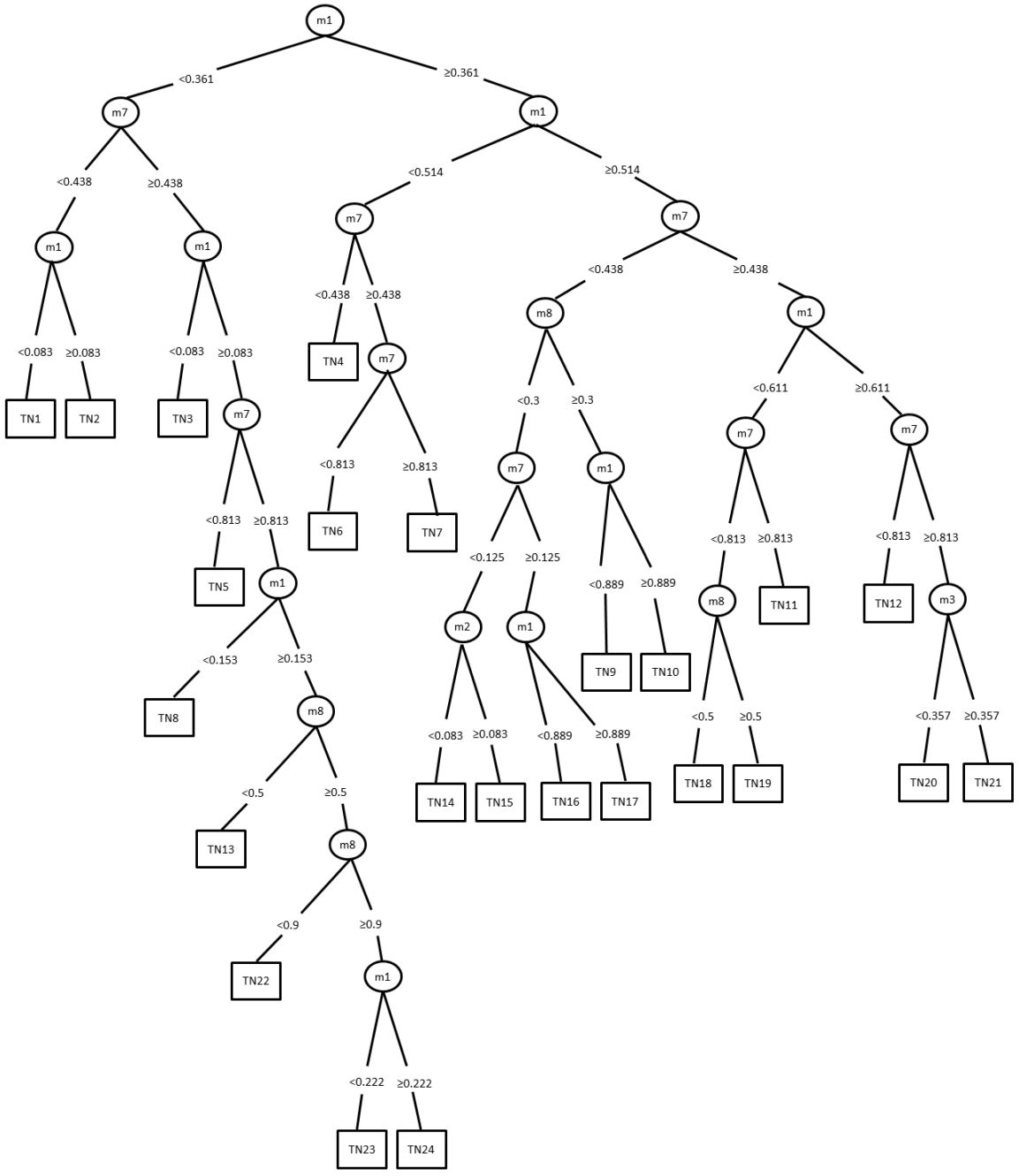

FIGURE 5.4: Constructed tree by M5' on Energy Efficiency Heating example.

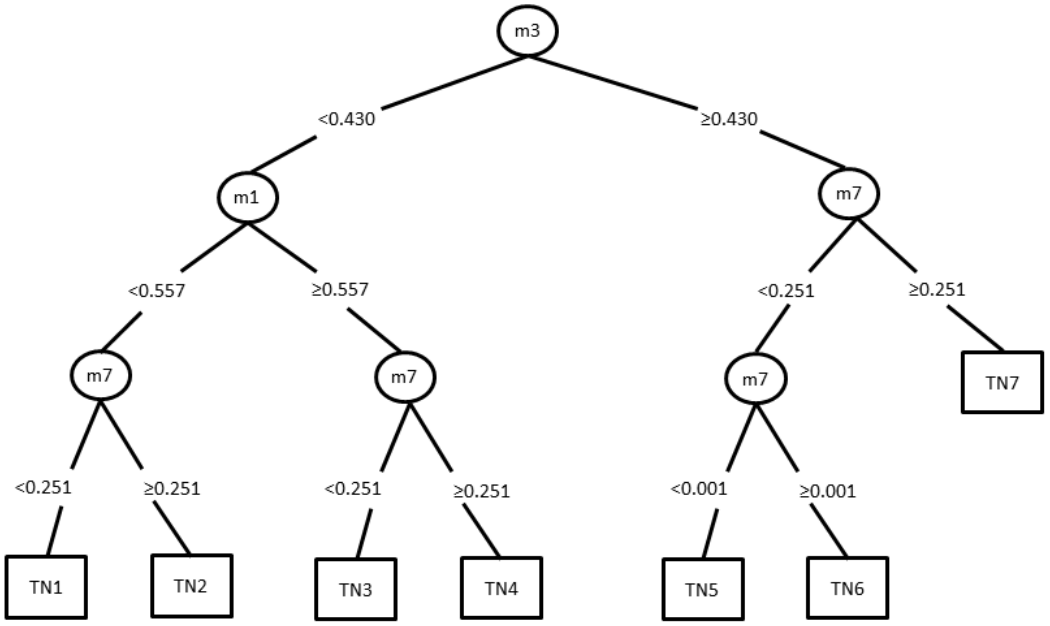

FIGURE 5.5: Constructed tree by ORTREE on Energy Efficiency Heating example.

According to Figure 5.3, CART has built a simple tree for the 768-sample example. On the top of the tree, CART splits the entire set of samples on feature *m1* at break-point of *0.361* into two child nodes, which are in turn further split on feature *m7* and *m1*, respectively. There are a total number of 7 terminal leaf nodes and the depth of the tree is equal to 4. From Figure 5.4, it is apparent that M5' has constructed a much larger tree than the CART. The top part of the M5' tree is almost identical as the tree built by CART, which is not surprising as the two algorithms share great similarity during tree growing procedure and only significantly different from each other on pruning procedure. Overall, the tree grown by M5' has a depth of 8 and 27 terminal leaf nodes, which is much harder to understand and interpret. Figure 5.5 visualises the actual tree built by our proposed ORTREE method. The size of the derived tree is similarly small as that of CART with 7 terminal leaf nodes and depth of 3, yet the two trees are

quite different as the root nodes of the two trees are split on different features. ORTREE, optimising the node splitting, picks feature *m3* as partition feature, in contrast to feature *m1* selected by CART. Overall on the Energy Efficiency Heating example, CART and ORTREE appear to build trees that are small in size, while M5' outputs a significantly larger tree.

The same analysis has been repeated on the other 5 benchmark datasets, and the results of which are available in Table 5.2. The same observation can be made that for the other examples, CART and ORTREE derive trees of similar numbers of terminal leaf nodes, while M5' sometimes builds trees of comparable sizes as the other two (i.e. Yacht Hydrodynamics and Concrete Strength) but more often outputs trees of several folds larger (i.e. Energy Efficiency Heating, Energy Efficiency Cooling, Airfoil and White Wine Quality).

TABLE 5.2: The number of terminal leaf nodes of the constructed trees by different regression tree learning methods

	Yacht Hydrodynamics	Concrete Strength	Energy Efficiency Heating	Energy Efficiency Cooling	Airfoil	White Wine Quality
CART	5	13	7	4	18	7
M5'	4	10	24	24	44	55
ORTREE	5	14	7	12	14	6

5.4 Concluding Remarks

Regression analysis is data-driven tool that aims to accurately predict a continuous output variable from a set of independent input variables. In this chapter, a novel solution procedure is proposed that extends the piece-wise linear regression method presented in the last chapter. The novel solution procedure recursively partitions the samples into two child nodes, with the OPLRA model being used to optimise the node binary splitting. The advantage of the novel solution procedure is that segmentation of multiple features is allowed, which is not possible for the previously proposed segmented regression method, which partitions one feature into multiple intervals. The novel solution procedure grows a regression tree model, named ORTREE.

A number of 6 benchmark datasets have been used to demonstrate the applicability and efficiency of the proposed ORTREE. Popular regression learning algorithms have been implemented for comparison, including tree-based CART, M5' and Cubist and methods based on other methodologies, including MARS, MLP, kriging, segmented regression, etc. The results of cross validation clearly indicate that ORTREE always result in lower prediction error than the segmented regression model. In terms of comparison against other methods, ORTREE offers consistently good predictive performance as it trails the accuracy of Cubist by very narrow margin while comprehensively outperforming other methods.

In the near future, a few aspects can be investigated in order to further refine the ORTREE method. Most of the existing regression tree learning algorithms, including the proposed ORTREE, perform binary splits recursively to keep the tree growing. Splitting a parent node into multiple child nodes, instead of two, is likely to better explore the structure of the dataset. In the proposed ORTREE, child nodes are fitted with multiple linear regression functions, which may be inadequate when the local relationship exhibits very strong non-linear relationship. More basis functions, for example quadratic, logarithmic terms can be easily added to the underlying ORTREE model so that the child nodes can model higher order non-linearity.

Chapter 6

Identifying Community Structure in Directed Networks Maximising Modularity

Network is a powerful means of representing various types of complex systems, where nodes denote the entities and edges define the existence of interactions between the entities. One important topological property of many real world networks is called *community structure*, where in various sub-graphs of a network, the density of edges within each subgraph is much higher than across different subgraphs. Each of those sub-graphs can be viewed as a community/module. In literature, a metric called modularity is defined that measures the quality of a partition of nodes into different mutually exclusive communities. One means of deriving community structure is modularity maximisation. In this chapter, a novel mathematical programming-based model is proposed that tackles the problem of maximising modularity for directed networks.

6.1 Introduction and Literature Review

It is increasingly clear that a wide range of systems across different disciplines can be described using network representations. The edges in a network can be binary/boolean, weighted, directed and with positive or negative sign, thus making network a suitable tool to model diverse types of interactions. One of the most

fundamental phenomenon observed from the graph analysis of those real world networks is that real world networks are not random graphs with homogeneous edge distribution. Rather, high density of edges exist within certain subsets of nodes, while the density of edges between those subsets of nodes are much lower, giving rise to the presence of strong community structure [240].

Uncovering the community structure in networks can provide insightful views into the systems under study, as each community can be viewed as an independent building block of nodes that are likely to share similar properties and/or collectively perform specific tasks. In protein interaction networks, proteins that have similar functions are more likely to be grouped into the same community [241]. Therefore, community detection can be used as a tool to predict unknown function of proteins by assigning them to modules of known functionality. Accurate identification of protein functional modules is also important for applications like drug design, where relevant modules of proteins can serve as potential drug targets [242]. In Web networks, Web pages deal with similar topics are clustered into the same modules [243]. Identifying community structures has been shown to be helpful in building powerful search engines. In scientific collaboration network, where researchers are nodes and an edge exists if two researches co-author a publication, communities structures are shown to be strong [56].

Informally, community detection refers to the procedure of identifying the inherent higher order structure of a network by partitioning nodes into different modules. A metric, called modularity (Q), is defined by Newman and Girvan [59] for undirected networks, which measures the quality of network divisions. Scaled between 0 and 1, modularity is computed as the number of edges placed within modules minus the expected number of edges that should fall within the modules in an equivalent network with randomly placed edges (null case). High modularity value indicates strong evidence that the given network division owns a statistically significantly larger proportion of edges within the modules than random, thus posing strong community structure. On the contrary, low modularity values (close to 0) suggest that the identified network division is not better than random. It is generally believed that a modularity value of more than 0.3 suggests significant underlying community structure.

With modularity being used as objective function, community detection can be

formulated as an optimisation problem of modularity maximisation. Modularity maximisation has been proven to be NP-complete [244], and therefore exhaustive search algorithm is only applicable to very small networks as the number of possible partitions increases at least exponentially with the number of nodes. Effective algorithms have been proposed in literature that are based on a number of methodologies, including simulated annealing [245, 246], greedy algorithms [59], extremal optimisation [247], spectral algorithm [248] and also integer programming-based optimisation model [58, 62].

While modularity optimisation for undirected network has been well studied over the past decade, little has been done for module detection in directed network. Many real world networks, however, are inherently directed, including World Wide Web networks where an edge represents a hyperlink from one page to another. In brain neural networks, directed connections between subjects denote the information flow [249]. In a metabolic network, directed edge represents material flow from one substrate to a product, which may be irreversible. In literature, the conventional manner of tackling the problem of community detection in directed networks is simply to discard the directionality of edges, treat the networks as undirected, and apply the methods described above [250]. Unfortunately, this approach essentially loses valuable information carried in the edge directions.

Leicht and Newman [57] generalises the original modularity for undirected network to modularity for directed network by explicitly considering the in-degree and out-degree distributions of node. In-degree and out-degree of a node respectively refers to the number of edges points into and from this node. A number of methods exist in literature that are either designed specifically for optimising directed modularity or originally proposed for undirected modularity but can accommodate directed modularity as well. Those methods are briefly summarised below before a novel optimisation framework is introduced in subsequent sections.

6.1.1 Tabu Search

In [67], the stochastic-based tabu search is adopted to optimise modularity for undirected network, and the method can be directly applied for modularity optimisation of directed network by only modifying the objective function of modularity formula. Starting from an initial network division, tabu search iteratively

updates the partition by exploring the neighbourhood space. During each iteration, tabu search picks one node at a time, randomly assigns it to one of the other modules or a new module and evaluate the new modularity value. The best solution is selected and passed to the next round of iteration. To help escape local optimality, for each iteration, a set of nodes being recently updated are prevented from moving. The set of frozen nodes is constantly updated so that a node added to the set at one point will be removed after certain number of iterations.

6.1.2 Extremal Optimisation

Duch and Areans [247] introduce a new division community detection method which is based on extremal optimisation. Firstly, a given network is randomly divided into two modules of the same number of nodes. An iterative procedure is employed to improve the binary partition. The procedure works by calculating the relative contribution of each node towards modularity value in the current partition and moving the node contributing least to the opposite community. After each movement of node, the relative contribution of nodes need to be re-computed, before the least contributing node is re-allocated to the other community. This procedure stops when no improvement is observed with node re-assignment. After that, the two modules are isolated, i.e. the edges connecting one module to the other are deleted, and the above procedure of random splitting and iterative node allocation is applied to each module in turn. The whole method terminates when no module division can further increase modularity.

6.1.3 Fast Algorithm

Newman proposes a greedy algorithm [60] aiming to tackle medium to large networks at small computational cost. The agglomerative hierarchical method starts by assigning each node to their own communities. Subsequently, two communities are repeatedly selected and merged to form a larger community. At each step of joining communities in pairs, all combinations of pair-wise communities are tested and the one yielding greatest improvement in modularity is selected. Among the series of divisions generated, the one with best modularity value is taken as the final solution. This method has been demonstrated to be computationally cheap for very large networks.

6.1.4 PageRank Random Walk

Different from the above algorithms who optimises the modularity for directed network directly, the method proposed in [251] uses a PageRank random walk to transfer a directed network into an undirected network. PageRank is used to evaluate the similarity between pair-wise nodes in the directed networks, explicitly taking into account the edge directionality. The similarity matrix can be interpreted as an undirected but weighted network, where the similarity corresponds to edge weights. At last, standard module detection methods presented for undirected network can be employed to identify communities.

In the following section, a mathematical programming-based optimisation framework is built as an alternative approach to the existing ones in literature. Firstly, a mixed integer non-linear programming model is proposed to tackle the problem of maximising modularity of directed network. The non-linear constraints are later linearised, which result in an MILP model. An efficient iterative solution procedure is introduced that, by repeatedly fixing the community memberships of certain nodes and allowing other nodes to move to other communities, reduces the problem of solving a large MILP into solving a series of small MILP problems. A number of directed networks are used to benchmark the performance of various methods.

6.2 A Mathematical Programming-based Optimisation Framework for Modularity Optimisation in Directed Networks

In this section, a novel approach is proposed for the problem of community detection in directed network via modularity optimisation. The proposed integer programming models and iterative solution procedure are described in detail below.

6.2.1 A Mixed Integer Non-linear Programming Model for Modularity Optimisation

The problem of maximising modularity for directed, and possibly weighted, networks can be conveniently formulated as an MINLP model. The indices, parameters and variables associated with the proposed model are listed below:

Indices	
n, e	node
m	module
l_{ne}	directed edge pointing from node n to e
Parameters	
β_{ne}	weight of edge point from node n to e
d_n^{in}	sum of weights over all edges points to node n; in-coming edge weight
d_n^{out}	sum of weights over all edges points from node n; out-going edge weight
L	total amount of weights over all edges in the given network
Binary variables	
Y_{nm}	*1* if node n belongs to module m; *0* otherwise
Free variables	
D_m^{in}	sum of d_n^{in} for all the nodes belong to module m ($Y_{nm} = 1$)
D_m^{out}	sum of d_n^{out} for all the nodes belong to module m ($Y_{nm} = 1$)
L_m	sum of edge weights in module m

In community detection, we are mainly concerned with hard parition, i.e. one node can only be allocated to exactly one module, which is modelled via the below constraints:

$$\sum_m Y_{nm} = 1 \qquad \forall n \tag{6.1}$$

where binary variables Y_{nm} are equal to *1* if node n is assigned into module m. For a given directed network, the sum of weights of edges coming into node n are denoted as parameter d_n^{in}, while the sum of weights of edges pointing from node n are d_n^{out}. For an unweighted network, d_n^{in} and d_n^{out} are respectively reduced to the in-degree and out-degree of node n. For a module m, the sum of d_n^{in} and d_n^{out} over

all nodes belonging to this module ($Y_{nm} = 1$) are respectively calculated as below:

$$D_m^{in} = \sum_n d_n^{in} Y_{nm} \qquad \forall n \tag{6.2}$$

$$D_m^{out} = \sum_n d_n^{out} Y_{nm} \qquad \forall n \tag{6.3}$$

For module m, the sum of weights of edges β_{ne} belonging to this module is computed as:

$$L_m = \sum_{n,e \in l_{ne}} \beta_{ne} Y_{nm} Y_{em} \qquad \forall m \tag{6.4}$$

In the above equations, an edge from node n to e is included in a module m if and only if both nodes n and e belong to module m, i.e. $Y_{nm} = 1, Y_{em} = 1$. Modularity is defined as the number of directed edges fall into communities minus the expected number of edges that should fall into communities in an null configuration of the equivalent network with edges being randomly placed [57]:

$$max \quad Q = \sum_m (\frac{L_m}{L} - \frac{D_m^{in} D_m^{out}}{L^2}) \tag{6.5}$$

The MINLP model is summarised as below:

Objective function (6.5)
Subject to:
One module for each node (6.1)
Sum of in-coming edge weights over all nodes in a module (6.2)
Sum of out-going edge weights over all nodes in a module (6.3)
Total edge weights in a module (6.4)
$D_m^{in}, D_m^{out}, L_m \geq 0$, $Y_{nm} \in \{0, 1\}$

The presence of non-linearity, combined with the use of integer variables present considerable computational difficulty for finding globally optimal solution. Solving MINLP problems typically involves repeatedly specifying different initial starting points and solving the model to identify locally optimal solutions, which can generally be realised in very affordable computational time. However quality of the solutions are hard to guarantee. Thus, this MINLP model can be used to provide an initial network division, before a more sophisticated method can be applied to

refine the division. In the next section, the MINLP model is reformulated to an MINLP by linearising the two non-linear constraints.

6.2.2 A Mixed Integer Linear Programming Model for Modularity Optimisation

The two non-linear constraints are exactly linearised, i.e. the linearised constraints are exact representations of the original non-linear constraints and therefore keeping the original decision space intact, leading to an MILP reformulation. Firstly, Equ. (6.4) can be replaced with the following three sets of constraints:

$$LS_{nem} \leq \beta_{ne} Y_{nm} \qquad \forall n, e \in l_{ne}, m \tag{6.6}$$

$$LS_{nem} \leq \beta_{ne} Y_{em} \qquad \forall n, e \in l_{ne}, m \tag{6.7}$$

$$L_m = \sum_{n, e \in l_{ne}} LS_{nem} \qquad \forall m \tag{6.8}$$

where LS_{nem} are newly introduced positive intermediate variables. For edge from node n to e, LS_{nem} is equal to its weight β_{ne} if it belongs to module m; *0* otherwise. The non-linear term in the objective function, $D_m^{in} D_m^{out}$, can be re-written as below:

$$\begin{aligned} D_m^{in} D_m^{out} &= D_m^{in} (\sum_n d_n^{out} Y_{nm}) \\ &= \sum_n d_n^{out} (D_m^{in} Y_{nm}) \end{aligned}$$

where $D_m^{in} Y_{nm}$ is the product of a continuous variable D_m^{in} and a binary variable Y_{nm}, and can be linearised using the following set of equations:

$$DY_{nm} \geq D_m^{in} - U(1 - Y_{nm}) \qquad \forall n, m \tag{6.9}$$

$$DD_m^{in\_out} \geq \sum_n d_n^{out} DY_{nm} \qquad \forall m \tag{6.10}$$

where DY_{nm} are introduced as new positive variables to replace $D_m^{in} Y_{nm}$, $DD_m^{in\_out}$ are introduced to replace $D_m^{in} D_m^{out}$ and U is an arbitrarily large number. The

objective function now becomes:

$$max \quad Q = \sum_{m} (\frac{L_m}{L} - \frac{DD_m^{in\_out}}{L^2}) \tag{6.11}$$

The final model, getting rid of all the non-linearity, is named Di_MOD (Directed MODularity) and contains new objective function (6.11) and constraints (6.1), (6.2) and (6.6)-(6.10). The MILP model can be solved to global optimality for small problems, but still consumes large computational resource for larger problems. Therefore, an iterative solution procedure is also derived in the next section, which achieves an improved solution quality via repeatedly solving reduced MILP models.

6.2.3 An Iterative Algorithm Improves the Quality of Network Division

As vast majority of the module detection methods in literature, we derive a new iterative solution procedure to improve the quality of the final network partition. The pseudocode of the iterative method is provided below:

An initial network partition is required as input to the proposed iterative algorithm. Although in general any community detection method in literature can be utilised to serve the purpose, we make use of the MINLP model presented earlier in this chapter. The MINLP model has the advantage of producing a coarse network division at small computational cost. Specifically, the MINLP model is solved N_{MINLP} times, each from a different random initial starting point. The solution corresponding to the highest modularity value is retained as the initial network division and passes to the iterative algorithm.

Given the initial network partition, a module is picked and $N_{relaxed}$ nodes are randomly selected from the module, whose community memberships are relaxed/deleted. For all other nodes in the network, their community memberships are fixed. The reduced Di_MOD is then solved maximising modularity by determining only the community memberships of the relaxed nodes. Any of the relaxed nodes can either stay in the current community, move to another existing community to a new community. Note that by controlling $N_{relaxed}$ to be reasonably small, the resulting

Iterative Algorithm for Identifying Network Division

1.) An initial network division need to be provided. Here we make use of the MINLP model proposed earlier in this chapter, although any other algorithm in literature can be employed as well. The MINLP model is repeatedly solved from different random initial solutions N_{MINLP} times. Each run outputs a different network division.

2.) From *1.)*, the solution giving highest modularity value is kept as the initial network division.

3.) Do steps *4.)-6.)* for each module.

4.) Randomly select $N_{relaxed}$ nodes in the current module and relax them by removing their node-module allocations (Y_{nm}). For all the other nodes in the network, their node-module allocations are fixed.

5.) Solve a reduced Di_MOD model maximising modularity and determining the community memberships of the relaxed nodes.

6.) Within the current module, each node must be relaxed once and only once. If there are still nodes yet to be relaxed in the current module, repeat *4.)* and *5.)*. Note that if the number of nodes haven't been relaxed in the current module is less than N_{res}, then select all of them for the next repetition of *4.)* and *5.)*.

7.) Do *3.)* for a total number of $N_{iteration}$ iterations.

8.) The solution at the end of the final iteration is taken as the final network division.

Note that N_{MINLP}, $N_{relaxed}$ and $N_{iteration}$ are all user-specific parameters.

reduced Di_MOD model is several orders of magnitude smaller than one complete Di_MOD model on the entire network. Each node within the current module must be relaxed once and only once. The procedure of random node selection and re-allocation continues until all the nodes within the current module have been relaxed. Note that the last repetition of the above procedure for each community may select less than $N_{relaxed}$ nodes, as total number of yet relaxed nodes is smaller than $N_{relaxed}$.

Perform the above procedure sequentially for each module in turn, which completes one round of iteration. The total number of iterations is controlled by $N_{iteration}$. The final solution at the end of the last iteration is taken as the output network division. N_{MINLP}, $N_{relaxed}$ and $N_{iteration}$ are all user-specific parameters.

Compared to solving one large Di_MOD model directly optimising the community memberships of all nodes simultaneously, solving a series of reduced Di_MOD has the advantage of reaching global optimality for each reduced problem in small computational cost. In the next section, a number of directed networks are used to demonstrate the applicability and efficiency of the proposed community detection algorithm.

6.3 Results and Discussion

In this section, performance of the community detection method proposed in this work is tested against other established methods in literature on 4 directed networks. A directed and weighted network representing neural network of *Caenorhabditis Elegans*, referred to as *C. Elegans* in this chapter, has been compiled in [252]. The network consists of a total number of 297 nodes and 2345 distinct weighted edges. The second network, Roget, contains 994 nodes and 5058 directed and unweighted connections. Roget network details cross-references between categories of English words, where one edge points from one category to another if a reference is provided to the latter among the words of the former. Roget network is downloaded from: http://vlado.fmf.uni-lj.si/pub/networks/data/dic/roget/Roget.htm. The other two networks, i.e. *Mycobacterium Tubecrulosis* and *Plasmodium Falciparum*, represent biological pathways at the molecular and cellular levels. Node entries include protein or DNA, while each edge denotes a certain type of physical interaction between two nodes, e.g. one node activates another. *Mycobacterium Tubecrulosis* and *Plasmodium Falciparum* respectively owns 194 nodes, 849 edges and 1390 nodes, 6497 edges. The above 4 networks are summarised in Table 6.1.

Table 6.1: Summary of benchmark directed networks

	number of nodes	number of directed edges	weighted or unweighted
C. Elegans	297	2345	weighted
Roget	994	5058	unweighted
Mycobacterium Tubecrulosis	194	849	unweighted
Plasmodium Falciparum	1390	6497	unweighted

A total number of 3 module detections methods are implemented for comparison, including tabu search [67], extremal optimisation [247] and fast algorithm [60]. All 3 methods are realised in the software Radatool (http://deim.urv.cat/ sergio.gomez/radatools.php). Heuristics-based tabu search and extremal optimisation have stochastic behaviours, which means multiple executions may result in different network divisions. Thus both methods are executed 10 times for each network, and the best network division is reported for comparison. It is important to stress that greater numbers of execution runs have been tried, which fail to improve the best solution. For the fast algorithm, its deterministic nature means only one run is required.

In terms of the proposed community detection approach, MINLP are solved using SBB solver 100 times ($N_{MINLP} = 100$). 60 nodes are relaxed for each solve of the reduced Di_MOD ($R_{relaxed} = 60$). Di_MOD is solved using Cplex solver with optimality gap set as 0, i.e. globally optimal solution is always achieved. For the number of iterations of the solution algorithm, 100 rounds are specified ($N_{iteration} = 100$). The deterministic nature of the proposed approach means only one single execution is required.

6.3.1 The Iterative Algorithm Improves the Quality of Network Division

Using the 4 benchmark directed networks, the effectiveness and efficiency of the iterative algorithm presented in the last section is tested. Table 6.2 provides the comparison between the modularity of the initial coarse network division provided by solving MINLP model and that of the final refined one identified by iterative algorithm.

Table 6.2: Modularity improvement achieved by iterative algorithm

	C. Elegans	Roget	*Mycobacterium Tubecrulosis*	*Plasmodium Falciparum*
Initial network division	0.4934	0.5209	0.4839	0.7044
Final network division	0.5076	0.5854	0.5073	0.7236
Percentage improvement	0.0288	0.1238	0.0486	0.0273

According to Table 6.2, the refined network division generally represents a noticeable improvement over the initial division. Roget is the network benefiting most from the iterative procedure which improves the modularity value of the community structure by more than 10 %. The iterative algorithm also successfully boosts modularity by nearly 5 % on *Mycobacterium Tubercrulosis*. On *C. Elegans* and *Plasmodium Falciparum*, improvements of just below 3% can be observed. It is stressed here that the initial network division is obtained by executing 100 runs of MINLP model from different random initial starting points, and increase the number of runs from 100 to 1000 only marginally increase the modularity value, and the extend of increase is ignorable compared with that of the iterative algorithm.

6.3.2 Comparative Results

After demonstrating the efficiency of the iterative algorithm, comparison is performed between the proposed approach and 3 widely employed methods in literature with respect to modularity values, the results of which is provided in Table 6.3.

TABLE 6.3: Comparative testing of different community detection methods on benchmark datasets

	C. Elegans	Roget	*Mycobacterium Tubecrulosis*	*Plasmodium Falciparum*
Tabu search	0.4398	0.4942	0.4294	0.6485
Extremal optimisation	0.4729	0.5503	0.4605	0.6664
Fast algorithm	0.5059	0.5561	0.4567	0.6846
Di_MOD	0.5076	0.5854	0.5073	0.7236

Table 6.3 offers concrete evidence that the proposed module detection optimisation approach in this work outperforms the 3 established methods in literature by clear margins. In *C. Elegans* network, the proposed method slightly outscores fast algorithm, with both achieving modularities of more than 0.5. The network divisions provided by extremal optimisation and tabu search respectively give modularities of around 0.47 and 0.44. The advantage of the proposed method is more obvious in the other 3 networks. In Roget example, the proposed approach identifies a community structure of modularity of 0.5854, with fast algorithm, extremal optimisation and tabu seach separately returns modularity values of 0.5561, 0.5503 and 0.4942. In the two biological directed networks, the margins between the proposed method and the second best, which in both cases is fast algorithm, are around 0.05. Overall, the novel methodology proposed here consistently and convincingly beat the other 3 methods widely employed in literature.

6.4 Concluding Remarks

This chapter addresses the problem of community structure detection in directed network. Network is an useful means to represent and study various complex systems. One interesting property observed in real world networks is the presence of community structure, where nodes in the network tend to cluster into several mutually exclusive tightly connected modules with higher within-module edge density than across-module edge density. A metric called modularity exists in literature that quantities the quality of division of nodes into communities, which essentially

transfers the problem of community detection into an optimisation problem of modularity maximisation.

While modularity optimisation has been extensively studied for undirected networks, there is little research effort to detect modules in directed networks. A mathematical programming-based optimisation approach has been introduced to fill the gap in literature. Modularity optimisation for directed network can be conveniently formulated as an MINLP model, which converge to locally optimal solutions quickly for even large networks. The MINLP model is then reformulated to an MILP model via linearisation, which can be solved to obtain globally optimal solution for small networks. The novel community detection method proposed in this chapter consists of two major steps, taking advantage of both models. Firstly, the MINLP model is solved to produce an initial coarse network division. Given the initial network division, the iterative algorithm works by repeatedly removing the community memberships of random sets of nodes, solving the reduced MILP model and re-allocating the relaxed nodes to communities.

Using 4 directed networks covering a wide range of node and edge sizes, the proposed iterative algorithm appears to considerably improve the quality of the initial coarse network partition. Compared with 3 popular community detection methods in literature, the proposed approach in this work is demonstrated to consistently identify the best network division giving largest modularity value. Another advantage of the proposed approach is its deterministic nature, which means multiple executions will desirably converge to the same network division.

Chapter 7

Conclusions and Future Work

This thesis has tackled several important problems in data mining, including multi-class data classification, disease classification, regression analysis and community detection maximising modularity. In this final chapter, we make a conclusion for the work presented early in the thesis before providing some research directions for the future work.

7.1 Concluding Remarks

In this doctoral thesis, mathematical programming models and heuristics-based solution procedures have been introduced for several topics in data mining.

Chapter 1 gives a general introduction for the 4 main topics of this thesis and mathematical programming optimisation techniques. The scope and overall structure of the thesis are also presented.

Chapter 2 addresses the problem of data classification. A classification model in literature has been adopted where two novel solution procedures have been proposed to construct more efficient classifiers. The first improvement updates the weight distribution of samples during the iterative training procedure and assigns higher weights to some misclassified samples in the last iteration. Using real world benchmark examples, the new classifier is demonstrated to almost always lead to better predictions from the original method and also outperforms other popular methods in literature. The second refinement has been introduced to reduce the

computational cost of the training process, by partitioning the samples into two disjoint regions and thus decomposing the whole problem into two sub-problems. This scheme is shown to decrease the computational cost by the orders of 1 to 2 magnitudes while generally maintaining the same level of prediction accuracies.

In Chapter 3, a special type of classification problem of disease classification is tackled, where the number of features far exceeds that of the samples. Making use of reliable extra biological information on pathway gene sets, a novel optimisation model is proposed which summarises the expression patterns of genes in a pathway into a new composite feature, called pathway activity. Pathway activity is inferred in a supervised manner by maximising its discriminative power. Using a large number of published datasets covering several complex diseases, it is shown that the novel pathway activity inference model results in consistently higher prediction rates than other competing methods in literature, for both binary and multi-class problems.

Chapter 4 deals with regression analysis. An optimisation is proposed that segments a given feature into multiple mutually exclusive intervals while simultaneously fits one distinct linear regression model per interval. A heuristic procedure is also introduced that determines the key partitioning feature among all and the number of intervals. Real world datasets have been employed to demonstrate that the proposed piece-wise linear regression model achieves robust performance and outperform several state-of-the-arts approaches based on other methodologies.

The regression method proposed in Chapter 4 is further generalised in Chapter 5. The refined method employs the underlying optimisation from the last Chapter but performs recursive binary partitions, therefore permitting segmenting more than one feature. The resulting method gives a tree-like structure. Using the same benchmark problems, it is shown that this refined method always provides better predictions than the original piece-wise linear regression model, and consistently matches the best regression tree method in the existing literature.

The problem of community detection in directed networks is investigated in Chapter 6. An MINLP model has firstly been introduced which partitions nodes in a network into clusters while maximising modularity value. Exact linearisation of non-linear constraints is then performed to produce a MILP model. A hybrid

method is formed by solving multiple times the MINLP models for identification of a good initial solution and then iteratively solving reduced MILP models to improve the initial network partition. With 4 case studies describing directed networks of various sizes, we show that this hybrid framework is capable of identifying network clusters that correspond to convincingly higher modularity values than other established methods.

A number of peer reviewed journal publications have arisen from the work of this thesis, which are listed in Appendix **??**.

7.2 Future Work

This section makes some suggestions on the future research directions of the topics this thesis has covered. For multi-class data classification, a natural extension of the work of this thesis is to generalise the data partitioning scheme so that more than 2 disjoint regions can be created, making it suitable for problems with large number of samples.

With regards to disease classifications, it is observed that the DIGS model proposed in this thesis is only able to achieve locally optimal solutions in feasible computational time. A possible avenue is to consider reformulating the current model so that some of the difficult binary variables can be replaced by easy continuous variables, thus potentially achieving a higher quality solution. On the other hand, disease classification can be approached by incorporating other biological knowledge, for example protein interaction network. Protein interaction network details general non-context specific gene-gene interactions, which can be integrated with pathway information to create pathway-specific gene-gene networks. Algorithms can be proposed which searches a small module in each pathway-specific network whose member genes can be combined into module activity for better classification.

For regression analysis, the current OPRLA model can be generalised to model higher order non-linearity by including more polynomial functions of the features. Furthermore, the proposed regression tree method can be extended to allow multiple instead of binary partitions per node.

In addition, one possible direction for future work of community detection is to extend the current work to perform soft partition, whereby each node can be allocated to more than one cluster. The motivation is that nodes are often members of more than one group in practical networks, for example a protein can belong to multiple protein complexes and a person can be present in more than one friend circles. Uncovering those structures can provide valuable insights into the system as the nodes participating in multiple clusters are likely to have special functions worth further attentions.

Lastly, the performance of a data mining method can be affected by parameter tuning to a considerable extent, especially for algorithms like neural network and support vector machine, where true optimal performance can only be found by carefully examining a wide range of parameter space. Therefore, it is of great interest as future work to thoughtfully compare the performance of the various proposed methods in this work against the existing ones in literature, by making parameter tuning an integral part of the training process.

Appendix A

Thesis Appendix A

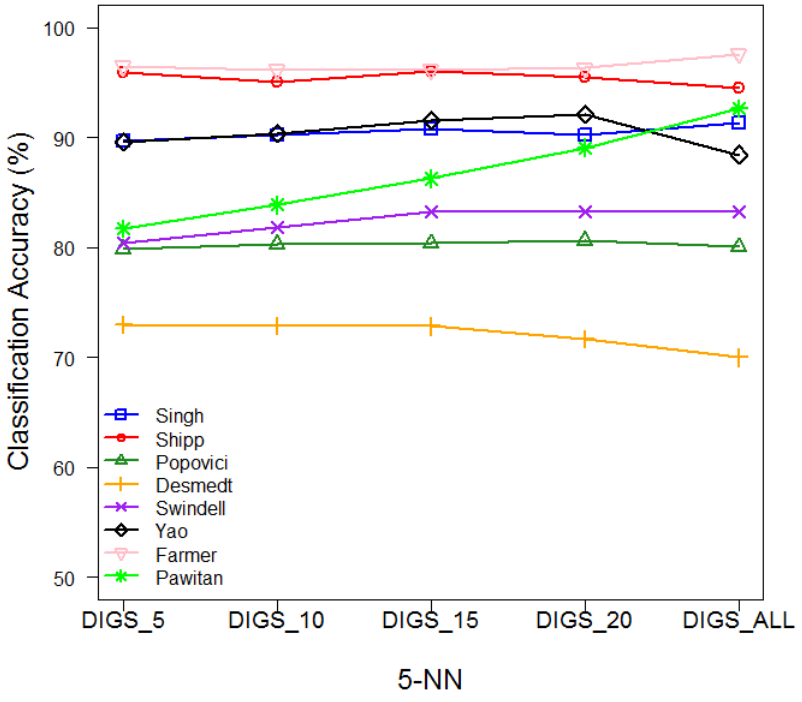

FIGURE A.1: Sensitivity analysis of parameter *NoG* for DIGS model with 5-NN. For each of the 8 datasets, the proposed DIGS model is applied to infer pathway activity while setting *NoG*, i.e. the maximum number of member genes in a pathway allowed to have non-zero weights, to *5*, *10*, *15* and *20*. In addition, DIGS model is also applied with *NoG* set to equal to the number of available member genes in a pathway, i.e. all member genes can take non-zero weights to construct pathway activity.

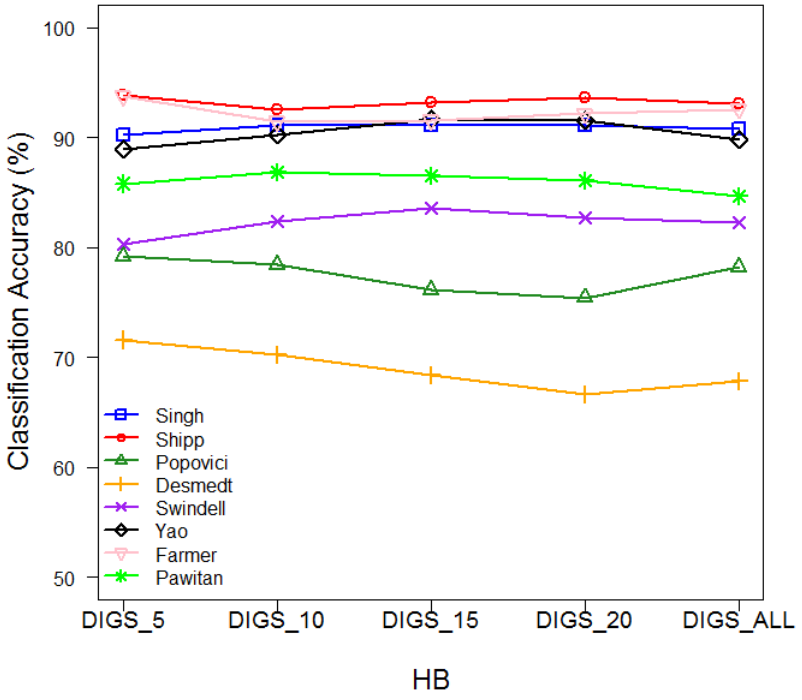

FIGURE A.2: Sensitivity analysis of parameter *NoG* for DIGS model with HB.

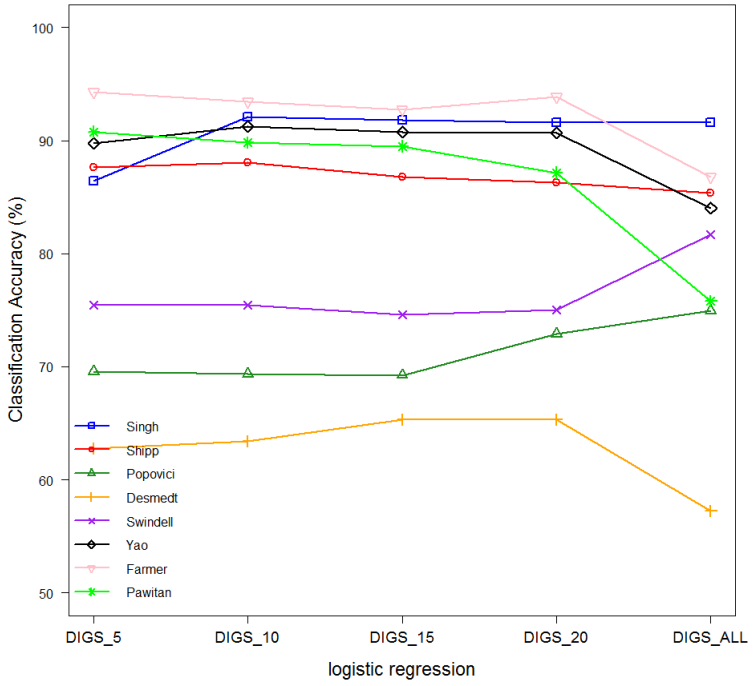

FIGURE A.3: Sensitivity analysis of parameter *NoG* for DIGS model with logistic regression classifier.

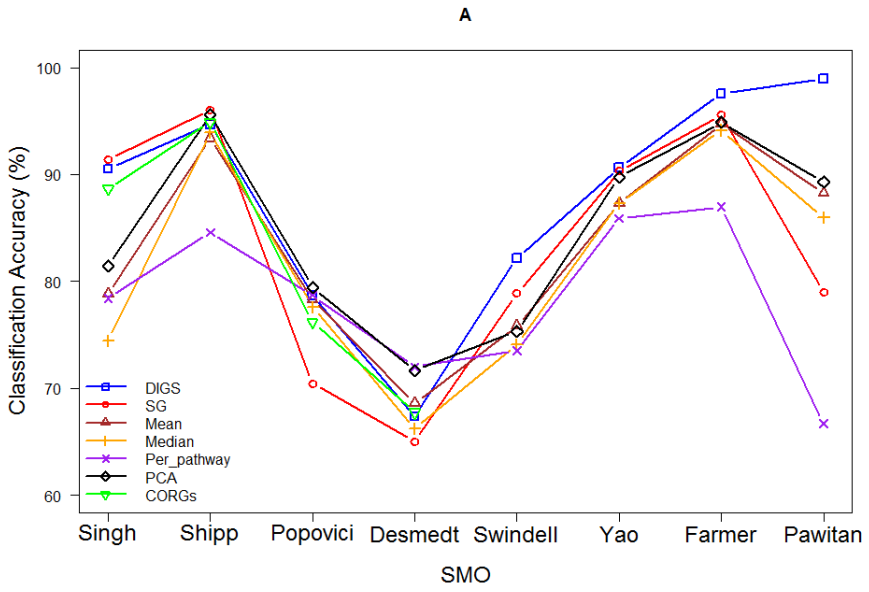

FIGURE A.4: Classification accuracy comparison of 7 competing methods using SMO classifier. The proposed DIGS pathway activity inference method is compared against other pathway activity inference methods (Mean, Median, PCA and CORGs) and also genes-based methods (SG and Per_pathway). Classification accuracy is summarised as average prediction rates over 50 runs of random partition of datasets into a 70% training set and a 30% testing set.

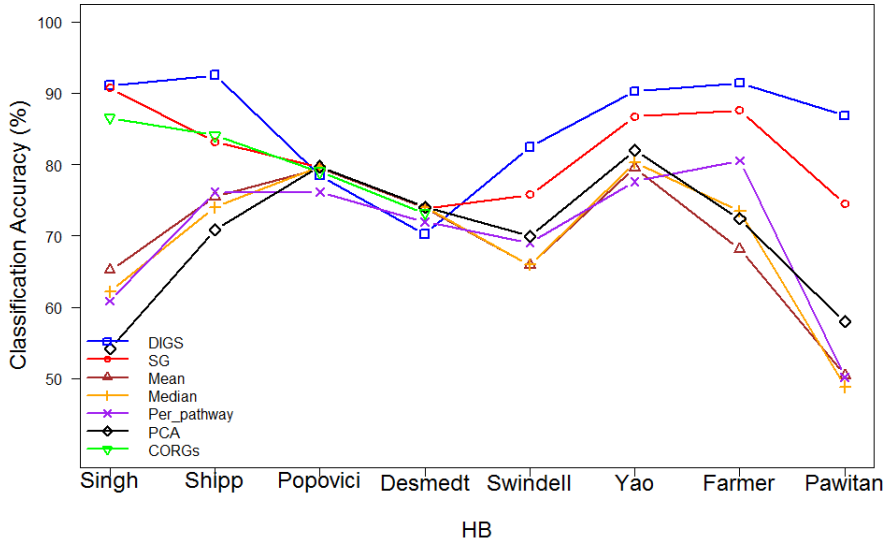

FIGURE A.5: Classification accuracy comparison of 7 competing methods using HB classifiers.

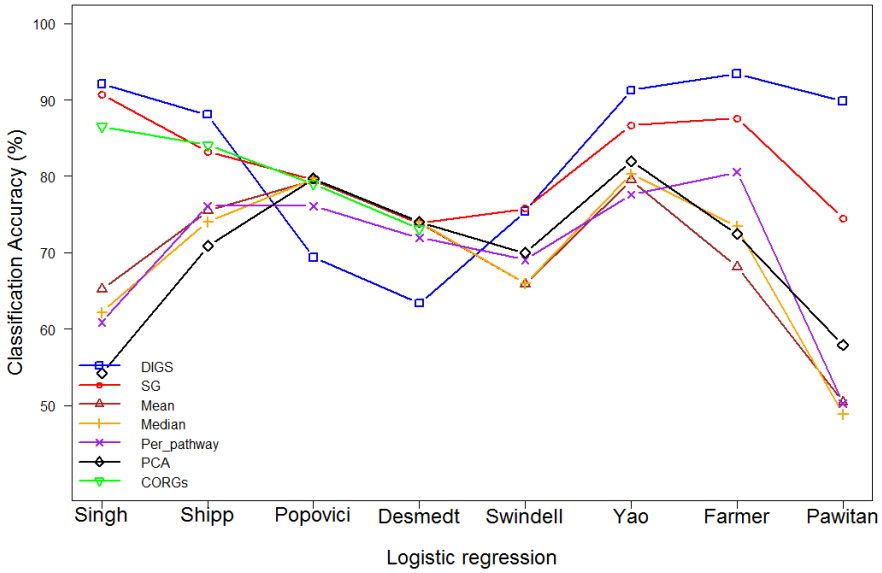

FIGURE A.6: Classification accuracy comparison of 7 competing methods using logistic regression classifier.

Appendix B

Thesis Appendix B

dataset	source	previous usage
Swindell [158]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE13355	[253], [254], [255]
Yao [34]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE14905	[256], [257], [258]
Farmer [159]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE1561	[259], [260], [261]
Pawitan [27]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE1456	[262], [263], [264]
Singh [113]	broadinstitute.org/publications/broad895	[265], [266], [267]
Shipp [33]	broadinstitute.org/cancer/software/genepattern/datasets	[268], [269], [270]
Popovici [115]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE24061	[271], [272], [273]
Desmedt [25]	ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7390	[274], [275], [276]

1		s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12
2	A1CF	6.299761	6.849294	7.112224	6.802398	6.741172	6.726306	6.646733	6.702756	6.630486	6.90861	6.757056	6.556576
3	A2M	10.55004	9.605868	9.640535	10.95427	11.06017	9.25476	10.57259	11.11637	10.78669	11.1757	10.29657	10.68608
4	A4GALT	5.764284	5.777656	6.120064	5.672856	5.832541	6.03651	5.902556	5.929536	6.054389	5.990723	6.273644	6.073681
5	A4GNT	6.385465	6.39045	6.459059	6.524506	6.49007	6.458041	6.313564	6.52101	6.471461	6.447539	6.508037	6.567449
6	AAAS	7.813748	8.289947	8.174065	7.85204	8.080942	8.167501	8.080862	8.071562	7.944216	8.329261	7.91028	8.139467
7	AACS	7.767451	8.105766	8.25626	7.333495	7.894891	9.045023	7.825295	7.745698	5.833531	8.326713	7.053235	8.473535
8	AADAC	4.890135	4.765803	5.019674	4.830515	4.99988	4.790212	4.764938	4.948455	4.791086	5.250301	4.892726	4.862714
9	AAGAB	8.674015	8.308887	9.126011	8.618939	8.741979	8.409913	8.915092	8.685316	8.212466	8.36833	7.727724	8.717949
10	AAK1	5.911435	5.991682	6.009859	6.144577	6.024232	6.034435	5.948805	5.820119	6.077574	6.234032	6.034787	5.784952
11	AAMP	8.160853	8.382912	7.93395	8.016574	8.285015	8.438293	8.264201	8.366192	7.775978	7.983289	7.989093	8.298759
12	AANAT	5.151494	5.245259	5.495521	5.203166	5.341063	5.280658	5.031006	5.354581	5.364189	5.159531	5.238636	5.284273
13	AARS	9.457041	9.709076	8.816787	9.206723	9.222702	9.681639	9.310491	9.226138	9.303415	9.379302	10.03477	9.252057
14	AARSD1	6.269374	6.117866	6.275916	5.689015	6.306557	6.253391	6.405225	6.331247	5.9336	6.012033	6.063138	6.249449
15	AASDHPPT	7.35948	7.677523	7.564451	7.229193	7.400607	8.445753	6.851547	7.7467	8.313728	7.118928	7.479454	7.944766
16	AASS	4.579639	4.428532	4.723596	4.647915	4.692669	4.618807	4.473132	4.627692	4.723413	4.836839	4.469412	4.486159
17	AATF	8.896249	9.053309	8.454489	8.479428	8.679761	9.240705	8.819105	8.82414	8.456685	9.340011	8.519816	8.633385
18	AATK	7.851845	8.224932	8.124996	7.346324	8.002528	7.550027	7.823015	7.393081	7.771715	7.854716	7.601578	7.412387
19	ABAT	6.677162	6.0157	8.738987	5.885649	7.181021	6.026451	7.158855	6.935308	4.98534	4.990128	4.928031	7.698374
20	ABCA1	7.174541	7.128298	6.314543	7.20193	7.077463	6.983015	7.296585	7.031339	7.129421	6.939336	7.232434	6.974238
21	ABCA11P	4.688826	4.645572	4.752203	4.873537	4.761736	4.770959	4.814431	4.597714	4.743846	4.903932	4.539537	4.600307
22	ABCA12	4.157606	6.42055	4.037078	4.199681	4.111424	3.890289	3.99441	4.082922	4.109193	4.832135	4.026585	4.093777
23	ABCA2	5.976609	6.323168	6.431118	6.090522	6.064238	5.943605	6.152053	6.050097	6.035087	6.267345	6.0819	5.966781

Figure B.1: Snapshot of part of Pawitan dataset.

1		s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12
2	A1CF	5.678048	5.990135	5.839396	6.572941	6.14092	6.234746	5.744836	6.17736	5.841815	6.160422	6.166071	5.971462
3	A2M	8.01091	9.119206	8.096509	7.95305	7.465575	10.58527	8.041107	9.89331	6.967598	7.138953	8.488835	7.305441
4	A4GALT	5.148008	5.229286	5.387221	5.397201	5.133244	5.379116	5.129882	4.978901	5.03611	5.851385	5.763482	5.411616
5	A4GNT	5.892704	6.004572	6.045526	6.12273	5.862248	6.265048	5.758454	6.046587	5.872133	6.093176	6.237637	6.135632
6	AAAS	7.508075	7.660116	7.655036	7.746964	7.482541	7.354806	7.544592	7.290631	7.501065	7.994078	7.654724	7.899679
7	AACS	7.783166	7.739571	8.303449	7.968043	11.43205	6.584212	7.632107	8.030813	8.20918	7.04845	6.043012	7.726237
8	AADAC	4.727911	4.665343	4.767168	4.794391	5.34429	5.042084	4.652108	4.964437	4.767588	5.028751	5.250841	4.749172
9	AAGAB	7.818336	6.637145	7.016713	7.030032	6.660477	6.497377	7.076872	6.535321	6.259138	6.359559	6.382409	6.794145
10	AAK1	5.479258	5.343607	5.307743	5.508593	5.491997	5.636606	5.265009	5.514898	5.267362	5.616941	5.533442	5.48936
11	AAMP	8.527535	8.631923	8.874884	8.445504	8.52608	8.107767	8.545053	8.351008	8.791733	8.40895	8.09387	7.962094
12	AANAT	4.603697	4.763305	4.589214	4.801446	4.697462	4.605361	4.496371	4.508498	4.478183	5.407781	5.288463	4.976336
13	AARS	8.530414	8.087249	8.327129	8.791761	9.26773	8.623307	9.32596	8.790383	8.782786	8.492026	7.993787	9.193969
14	AARSD1	6.045628	6.087204	5.490916	6.196046	5.429548	5.879999	5.963912	5.800823	5.494524	6.045942	5.617109	5.936834
15	AASDHPPT	6.471591	7.461625	6.537777	6.487112	6.863911	6.712932	7.075273	6.745055	7.67735	5.215187	5.700437	6.693295
16	AASS	4.093751	4.005656	3.866538	4.016227	3.931816	3.98411	3.866167	3.92641	3.938082	4.181497	3.915055	3.882605
17	AATF	7.644809	7.477151	8.351854	8.337586	6.77835	7.139243	8.036817	7.23426	6.702748	7.260414	7.422823	7.862428
18	AATK	7.198725	7.5158	7.081631	7.801268	7.229595	7.355414	6.972929	7.264785	7.45741	7.721018	7.86395	7.542632
19	ABAT	4.664913	6.05537	6.322713	5.217686	4.419807	4.567428	5.181243	4.744039	4.020268	5.617494	5.010346	4.934933
20	ABCA1	5.467598	5.543501	5.539356	5.550424	5.398975	5.899353	5.243443	6.121227	5.21907	5.364465	5.777638	5.151225
21	ABCA11P	4.060006	4.15264	4.038197	4.273133	3.994413	4.190406	4.117449	4.274325	4.04415	4.283252	4.348432	4.174107
22	ABCA12	4.117277	3.738788	3.687677	3.698503	5.069442	3.956052	3.85028	4.531808	3.823266	3.964631	3.832543	4.288705
23	ABCA2	5.601484	5.514632	5.587829	5.83066	5.243851	5.51189	5.887195	5.42055	5.480731	5.835071	5.647872	5.538858

FIGURE B.2: Snapshot of part of Popovici dataset.

Appendix C

Thesis Appendix C

Lingjian Yang, Chrysanthi Ainali, Aristotelis Kittas, Frank O. Nestle, Lazaros G. Papageorgiou, Sophia Tsoka. Pathway-level disease data mining through hyper-box principles. *Mathematical Biosciences*, 260:25-34, 2014.

Lingjian Yang, Chrysanthi Ainali, Sophia Tsoka, Lazaros G. Papageorgiou. Pathway activity inference for multiclass disease classification through a mathematical programming optimisation framework. *BMC Bioinformatics*, 15:390, 2014.

Lingjian Yang, Songsong Liu, Sophia Tsoka, Lazaros G. Papageorgiou. Sample re-weighting hyper box classifier for multi-class data classification. *Computers & Industrial Engineering*, 85:44-56, 2015.

Lingjian Yang, Songsong Liu, Sophia Tsoka, Lazaros G. Papageorgiou. Mathematical programming for piecewise linear regression analysis. *Expert Systems with Applications*, 44:156-167, 2016.

Lingjian Yang, Songsong Liu, Sophia Tsoka, Lazaros G. Papageorgiou. Mathematical programming building regression tree model. *Journal of Machine Learning Research*, under review.

Bibliography

[1] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[2] David J. Hand, Padhraic Smyth, and Heikki Mannila. *Principles of Data Mining*. MIT Press, Cambridge, MA, USA, 2001.

[3] Shiguo Wang. A comprehensive survey of data mining-based accounting-fraud detection research. In *Intelligent Computation Technology and Automation (ICICTA), 2010 International Conference on*, volume 1, pages 50–53, 2010.

[4] Huseyin Ince and Bora Aktan. A comparison of data mining techniques for credit scoring in banking: A managerial perspective. *Journal of Business Economics and Management*, 10(3):233–240, 2009.

[5] Marc J. van de Vijver, Yudong D. He, Laura J. van 't Veer, Hongyue Dai, Augustinus A.M. Hart, Dorien W. Voskuil, George J. Schreiber, Johannes L. Peterse, Chris Roberts, Matthew J. Marton, Mark Parrish, Douwe Atsma, Anke Witteveen, Annuska Glas, Leonie Delahaye, Tony van der Velde, Harry Bartelink, Sjoerd Rodenhuis, Emiel T. Rutgers, Stephen H. Friend, and René Bernards. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347(25):1999–2009, 2002.

[6] Yixin Wang, Jan GM Klijn, Yi Zhang, Anieta M Sieuwerts, Maxime P Look, Fei Yang, Dmitri Talantov, Mieke Timmermans, Marion E Meijer van Gelder, Jack Yu, Tim Jatkoe, Els MJJ Berns, David Atkins, and John A Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *The Lancet*, 365(9460):671 – 679, 2005.

[7] James C. Costello, Laura M. Heiser, Elisabeth Georgii, Mehmet Gönen, Michael P. Menden, Nicholas J. Wang, Mukesh Bansal, Muhammad Ammad-ud din, Petteri Hintsanen, Suleiman A. Khan, John-patrick Mpindi, Olli Kallioniemi, Antti Honkela, Tero Aittokallio, Krister Wennerberg, James J. Collins, Dan Gallahan, Dinah Singer, Julio Saez-rodriguez, Samuel Kaski, Joe W. Gray, and Gustavo Stolovitzky. A community effort to assess and improve drug sensitivity prediction algorithms. *Nature biotechnology*, 32(12): 1202–12, 2014.

[8] Lisa Hellerstein. Machine learning: A theoretical approach by balas k. natarajan. morgan kaufmann publishers, inc., 1991. *Machine Learning*, 13 (1):145–149, 1993.

[9] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[10] C. Radhakrishna Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2):pp. 159–203, 1948.

[11] Guobin Ou and Yi Lu Murphey. Multi-class pattern classification using neural networks. *Pattern Recognition*, 40(1):4 – 18, 2007.

[12] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2): 415–425, 2002.

[13] B.B. Chaudhuri and U. Bhattacharya. Efficient training and improved performance of multilayer perceptron in pattern classification. *Neurocomputing*, 34(1–4):11 – 27, 2000.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press, Cambridge, MA, USA, 1986.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[16] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? In Catriel Beeri and Peter Buneman, editors, *Database Theory — ICDT'99*, volume 1540 of *Lecture Notes in Computer Science*, pages 217–235. Springer Berlin Heidelberg, 1999.

[17] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, 2009.

[18] Jerome H. Friedman. Multivariate adaptive regression splines. *Ann. Statist.*, 19(1):1–67, 1991.

[19] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[20] Alin Dobra and Johannes Gehrke. Secret: A scalable linear regression tree algorithm. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 481–487, New York, NY, USA, 2002. ACM.

[21] Robi Polikar. Ensemble learning. In Cha Zhang and Yunqian Ma, editors, *Ensemble Machine Learning*, pages 1–34. Springer US, 2012.

[22] Therese Sørlie, Charles M. Perou, Robert Tibshirani, Turid Aas, Stephanie Geisler, Hilde Johnsen, Trevor Hastie, Michael B. Eisen, Matt van de Rijn, Stefanie S. Jeffrey, Thor Thorsen, Hanne Quist, John C. Matese, Patrick O. Brown, David Botstein, Per Eystein Lønning, and Anne-Lise Børresen-Dale. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, 98(19):10869–10874, 2001.

[23] Jorge S Reis-Filho and Lajos Pusztai. Gene expression profiling in breast cancer: classification, prognostication, and prediction. *The Lancet*, 378 (9805):1812 – 1823, 2011.

[24] Laura van 't Veer, Hongyue Dai, Marc van de Vijver, Yudong He, Augustinus Hart, Rene Bernards, and Stephen Friend. Expression profiling predicts outcome in breast cancer. *Breast Cancer Res*, 5(1):57–58, 2003.

[25] Christine Desmedt, Fanny Piette, Sherene Loi, Yixin Wang, Françoise Lallemand, Benjamin Haibe-Kains, Giuseppe Viale, Mauro Delorenzi, Yi Zhang,

Mahasti Saghatchian d'Assignies, Jonas Bergh, Rosette Lidereau, Paul Ellis, Adrian L. Harris, Jan G.M. Klijn, John A. Foekens, Fatima Cardoso, Martine J. Piccart, Marc Buyse, and Christos Sotiriou. Strong time dependence of the 76-gene prognostic signature for node-negative breast cancer patients in the transbig multicenter independent validation series. *Clinical Cancer Research*, 13(11):3207–3214, 2007.

[26] Archie Bleyer and H. Gilbert Welch. Effect of three decades of screening mammography on breast-cancer incidence. *New England Journal of Medicine*, 367(21):1998–2005, 2012.

[27] Yudi Pawitan, Judith Bjöhle, Lukas Amler, Anna-Lena Borg, Suzanne Egyhazi, Per Hall, Xia Han, Lars Holmberg, Fei Huang, Sigrid Klaar, EdisonT Liu, Lance Miller, Hans Nordgren, Alexander Ploner, Kerstin Sandelin, PeterM Shaw, Johanna Smeds, Lambert Skoog, Sara Wedrén, and Jonas Bergh. Gene expression profiling spares early breast cancer patients from adjuvant therapy: derived and validated in two population-based cohorts. *Breast Cancer Research*, 7(6):R953, 2005.

[28] John Quackenbush. Microarray analysis and tumor classification. *New England Journal of Medicine*, 354(23):2463–2472, 2006.

[29] Gavin Sherlock. Analysis of large-scale gene expression data. *Current Opinion in Immunology*, 12(2):201 – 205, 2000.

[30] T. Sorlie, M. van de Rijn, O. Fluge, Charles Perou, Therese Sørlie, Michael Eisen, Hilde Johnsen, Stefanie Jeffrey, Christian Rees, Jonathan Pollack, Douglas Ross, Lars Akslen, Øystein Fluge, Alexander Pergamenschikov, Cheryl Williams, Shirley Zhu, Per Lønning, Anne-Lise Børresen Dale, Patrick Brown, and David Botstein. Molecular portraits of human breast tumours. *Nature*, 406(6797), 2000.

[31] Laura J. Van 't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A. M. Hart, Mao Mao, Hans L. Peterse, Karin van der Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, Rene Bernards, and Stephen H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[32] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[33] Margaret A. Shipp, Ken N. Ross, Pablo Tamayo, Andrew P. Weng, Jeffery L. Kutok, Ricardo C. T. Aguiar, Michelle Gaasenbeek, Michael Angelo, Michael Reich, Geraldine S. Pinkus, Tane S. Ray, Margaret A. Koval, Kim W. Last, Andrew Norton, T. Andrew Lister, Jill Mesirov, Donna S. Neuberg, Eric S. Lander, Jon C. Aster, and Todd R. Golub. Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature Medicine*, 8(1):68–74, 2002.

[34] Yihong Yao, Laura Richman, Chris Morehouse, Melissa de los Reyes, Brandon W. Higgs, Anmarie Boutrin, Barbara White, Anthony Coyle, James Krueger, Peter A. Kiener, and Bahija Jallal. Type i interferon: Potential therapeutic target for psoriasis? *PLoS ONE*, 3(7):e2737, 2008.

[35] David G. Beer, Sharon L. R. Kardia, Chiang-Ching Huang, Thomas J. Giordano, Albert M. Levin, David E. Misek, Lin Lin, Guoan Chen, Tarek G. Gharib, Dafydd G. Thomas, Michelle L. Lizyness, Rork Kuick, Satoru Hayasaka, Jeremy M. G. Taylor, Mark D. Iannettoni, Mark B. Orringer, and Samir Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8(8):816–824, 2002.

[36] Liat Ein-Dor, Itai Kela, Gad Getz, David Givol, and Eytan Domany. Outcome signature genes in breast cancer: is there a unique set? *Bioinformatics*, 21(2):171–178, 2005.

[37] Liat Ein-Dor, Or Zuk, and Eytan Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proceedings of the National Academy of Sciences*, 103(15):5923–5928, 2006.

[38] David Venet, Jacques E. Dumont, and Vincent Detours. Most random gene expression signatures are significantly associated with breast cancer outcome. *PLoS Comput Biol*, 7(10):e1002240, 2011.

[39] Minoru Kanehisa and Susumu Goto. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30, 2000.

[40] Yotam Drier, Michal Sheffer, and Eytan Domany. Pathway-based personalized analysis of cancer. *Proceedings of the National Academy of Sciences*, 110(16):6388–6393, 2013.

[41] Eunjung Lee, Han-Yu Chuang, Jong-Won Kim, Trey Ideker, and Doheon Lee. Inferring pathway activity toward precise disease classification. *PLoS Comput Biol*, 4(11):e1000217, 2008.

[42] Han-Yu Chuang, Eunjung Lee, Yu-Tsueng Liu, Doheon Lee, and Trey Ideker. Network-based classification of breast cancer metastasis. *Molecular Systems Biology*, 3(1), 2007.

[43] Trey Ideker, Janusz Dutkowski, and Leroy Hood. Boosting signal-to-noise in complex biology: Prior knowledge is power. *Cell*, 144(6):860 – 863, 2011.

[44] Jianzhen Xu and Yongjin Li. Discovering disease-genes by topological features in human protein–protein interaction network. *Bioinformatics*, 22(22): 2800–2805, 2006.

[45] Lingjian Yang, Chrysanthi Ainali, Aristotelis Kittas, Frank O. Nestle, Lazaros G. Papageorgiou, and Sophia Tsoka. Pathway-level disease data mining through hyper-box principles. *Mathematical Biosciences*, 260(0):25 – 34, 2015.

[46] TaeJin Ahn, Eunjin Lee, Nam Huh, and Taesung Park. Personalized identification of altered pathways in cancer using accumulated normal tissue data. *Bioinformatics*, 30(17):i422–i429, 2014.

[47] Lu Tian, Steven A. Greenberg, Sek Won Kong, Josiah Altschuler, Isaac S. Kohane, and Peter J. Park. Discovering statistically significant pathways in expression profiling studies. *Proceedings of the National Academy of Sciences of the United States of America*, 102(38):13544–13549, 2005.

[48] Ke-Qin Liu, Zhi-Ping Liu, Jin-Kao Hao, Luonan Chen, and Xing-Ming Zhao. Identifying dysregulated pathways in cancers from pathway interaction networks. *BMC Bioinformatics*, 13(1):126, 2012.

[49] Rui Liu, Xiangdong Wang, Kazuyuki Aihara, and Luonan Chen. Early diagnosis of complex diseases by molecular biomarkers, network biomarkers, and dynamical network biomarkers. *Medicinal Research Reviews*, 34(3):455–478, 2014.

[50] JonM. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and AndrewS. Tomkins. The web as a graph: Measurements, models, and methods. In Takano Asano, Hideki Imai, D.T. Lee, Shin-ichi Nakano, and Takeshi Tokuyama, editors, *Computing and Combinatorics*, volume 1627 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 1999.

[51] David Lusseau. The emergent properties of a dolphin social network. *Proceedings of the Royal Society of London B: Biological Sciences*, 270(Suppl 2):S186–S188, 2003.

[52] David Lusseau and M. E. J. Newman. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London B: Biological Sciences*, 271(Suppl 6):S477–S481, 2004.

[53] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L. Barabasi. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[54] Victor Spirin, Mikhail S Gelfand, Andrey A Mironov, and Leonid A Mirny. A metabolic network in the evolutionary context: multiscale structure and modularity. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8774—8779, June 2006.

[55] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[56] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America*, 98(2):pp. 404–409, 2001.

[57] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Phys. Rev. Lett.*, 100:118703, 2008.

[58] Gang Xu, Laura Bennett, Lazaros Papageorgiou, and Sophia Tsoka. Module detection in complex networks using integer optimisation. *Algorithms for Molecular Biology*, 5(1):36, 2010.

[59] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004.

[60] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, 2004.

[61] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111, 2004.

[62] G. Xu, S. Tsoka, and L. G. Papageorgiou. Finding community structures in complex networks using mixed integer optimisation. *The European Physical Journal B*, 60(2):231–239, 2007.

[63] Ludo Waltman and NeesJan van Eck. A smart local moving algorithm for large-scale modularity-based community detection. *The European Physical Journal B*, 86(11):471, 2013.

[64] Chun-Cheng Lin, Jia-Rong Kang, and Jyun-Yu Chen. An integer programming approach and visual analysis for detecting hierarchical community structures in social networks. *Information Sciences*, 299(0):296 – 311, 2015.

[65] M. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[66] Fragkiskos D Malliaros and Michalis Vazirgiannis. Clustering and Community Detection in Directed Networks: A Survey. Technical Report arXiv:1308.0971, 2013.

[67] A Arenas, A Fernández, and S Gómez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[68] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72:027104, 2005.

[69] H. P. Williams. *Model Building in Mathematical Programming, 4th Edition.* Wiley, 4 edition, 1999.

[70] GAMS Development Corporation. *General Algebraic Modeling System (GAMS) Release 24.2.1.* Washington, DC, USA, 2013.

[71] Johannes Bisschop and Marcel Roelofs. *AIMMS - User's Guide.* Lulu.com, 2006. ISBN 1411698983.

[72] James Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Mathematical Programming*, 40(1-3):59–93, 1988.

[73] Margaret H. Wright. Interior methods for constrained optimization. *Acta Numerica*, 1:341–407, 1992.

[74] Gang Xu and Lazaros G. Papageorgiou. A mixed integer optimisation model for data classification. *Computers and Industrial Engineering*, 56(4):1205 – 1215, 2009.

[75] Toshiyuki Sueyoshi. Mixed integer programming approach of extended dea–discriminant analysis. *European Journal of Operational Research*, 152 (1):45 – 55, 2004.

[76] Jae H. Min and Young-Chan Lee. Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications*, 28(4):603 – 614, 2005.

[77] Toshiyuki Sueyoshi and Mika Goto. Dea–da for bankruptcy-based performance assessment: Misclassification analysis of japanese construction industry. *European Journal of Operational Research*, 199(2):576 – 594, 2009.

[78] Mingmin Chi, Rui Feng, and Lorenzo Bruzzone. Classification of hyperspectral remote-sensing data with primal {SVM} for small-sized training dataset problem. *Advances in Space Research*, 41(11):1793 – 1799, 2008.

[79] Santiago Velasco-Forero and Jesus Angulo. Classification of hyperspectral images by tensor modeling and additive morphological decomposition. *Pattern Recognition*, 46(2):566 – 577, 2013.

[80] David G. Lowe. Local naive bayes nearest neighbor for image classification. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 3650–3656, Washington, DC, USA, 2012. IEEE Computer Society.

[81] Onur Dagliyan, Fadime Uney-Yuksektepe, I. Halil Kavakli, and Metin Turkay. Optimization based tumor classification from microarray gene expression data. *PLoS ONE*, 6(2):e14579, 2011.

[82] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[83] Manjeevan Seera and Chee Peng Lim. A hybrid intelligent system for medical data classification. *Expert Systems with Applications*, 41(5):2239 – 2249, 2014.

[84] S. Anbazhagan and N. Kumarappan. A neural network approach to day-ahead deregulated electricity market prices classification. *Electric Power Systems Research*, 86(0):140 – 150, 2012.

[85] Wen Li, Duoqian Miao, and Weili Wang. Two-level hierarchical combination method for text classification. *Expert Systems with Applications*, 38(3):2030 – 2039, 2011.

[86] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[87] Hasan Bal and H. Hasan Örkcü. A new mathematical programming approach to multi-group classification problems. *Computers and Operations Research*, 38(1):105 – 111, 2011.

[88] Wang Zhen, Chen Jin, and Qin Ming. Non-parallel planes support vector machine for multi-class classification. In *Logistics Systems and Intelligent Management, 2010 International Conference on*, volume 1, pages 581–585, 2010.

[89] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2): 415–425, 2002.

[90] Laura Dioşan, Alexandrina Rogozan, and Jean-Pierre Pecuchet. Improving classification performance of support vector machine by genetically optimising kernel shape and hyper-parameters. *Applied Intelligence*, 36(2):280–294, 2012.

[91] Sedat Ozer, Chi H. Chen, and Hakan A. Cirpan. A set of new chebyshev kernel functions for support vector machine pattern classification. *Pattern Recognition*, 44(7):1435 – 1447, 2011.

[92] Taskin Kavzoglu. Increasing the accuracy of neural network classification using refined training data. *Environmental Modelling and Software*, 24(7): 850 – 858, 2009.

[93] Ganesh Arulampalam and Abdesselam Bouzerdoum. A generalized feedforward neural network architecture for classification and regression. *Neural Networks*, 16(5–6):561 – 568, 2003.

[94] D. Hunter, Hao Yu, M.S. Pukish, J. Kolbusz, and B.M. Wilamowski. Selection of proper neural network sizes and architectures-a comparative study. *Industrial Informatics, IEEE Transactions on*, 8(2):228–240, 2012.

[95] M. Martinez-Arroyo and L.E. Sucar. Learning an optimal naive bayes classifier. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1236–1239, 2006.

[96] TiagoA. Almeida, Jurandy Almeida, and Akebo Yamakami. Spam filtering: how the dimensionality reduction affects the accuracy of naive bayes classifiers. *Journal of Internet Services and Applications*, 1(3):183–200, 2011.

[97] Tzu-Tsung Wong. A hybrid discretization method for naïve bayesian classifiers. *Pattern Recognition*, 45(6):2321 – 2325, 2012.

[98] J. Brian Gray and Guangzhe Fan. Classification tree analysis using {TARGET}. *Computational Statistics and Data Analysis*, 52(3):1362 – 1372, 2008.

[99] Rajeev Rastogi and Kyuseok Shim. Public: A decision tree classifier that integrates building and pruning. *Data Min. Knowl. Discov.*, 4(4):315–344, 2000.

[100] William V. Gehrlein. General mathematical programming formulations for the statistical classification problem. *Oper. Res. Lett.*, 5(6):299–304, 1986.

[101] Hong Seo Ryoo. Pattern classification by concurrently determined piecewise linear and convex discriminant functions. *Computers and Industrial Engineering*, 51(1):79 – 89, 2006.

[102] A.M. Bagirov, J. Ugon, and D. Webb. An efficient algorithm for the incremental construction of a piecewise linear classifier. *Information Systems*, 36 (4):782 – 790, 2011.

[103] Dimitris Bertsimas and Romy Shioda. Classification and regression via integer optimization. *Oper. Res.*, 55(2):252–271, 2007.

[104] Alaleh Maskooki. Improving the efficiency of a mixed integer linear programming based approach for multi-class classification problem. *Computers and Industrial Engineering*, 66(2):383 – 388, 2013.

[105] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[106] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2): 105–139, 1999.

[107] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.

[108] Bing Niu, Yuhuan Jin, WenCong Lu, and GuoZheng Li. Predicting toxic action mechanisms of phenols using adaboost learner. *Chemometrics and Intelligent Laboratory Systems*, 96(1):43 – 48, 2009.

[109] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[110] Jan Adem and Willy Gochet. Mathematical programming based heuristics for improving lp-generated classifiers for the multiclass supervised classification problem. *European Journal of Operational Research*, 168(1):181 – 199, 2006.

[111] Kim Fung Lam and Jane W. Moy. Combining discriminant methods in solving classification problems in two-group discriminant analysis. *European Journal of Operational Research*, 138(2):294 – 301, 2002.

[112] Jun Luo, David J. Duggan, Yidong Chen, Jurga Sauvageot, Charles M. Ewing, Michael L. Bittner, Jeffrey M. Trent, and William B. Isaacs. Human prostate cancer and benign prostatic hyperplasia: Molecular dissection by gene expression profiling. *Cancer Research*, 61(12):4683–4688, 2001.

[113] Dinesh Singh, Phillip G. Febbo, Kenneth Ross, Donald G. Jackson, Judith Manola, Christine Ladd, Pablo Tamayo, Andrew A. Renshaw, Anthony V. D'Amico, Jerome P. Richie, Eric S. Lander, Massimo Loda, Philip W. Kantoff, Todd R. Golub, and William R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1(2):203 – 209, 2002.

[114] Christos Sotiriou, Pratyaksha Wirapati, Sherene Loi, Adrian Harris, Steve Fox, Johanna Smeds, Hans Nordgren, Pierre Farmer, Viviane Praz, Benjamin Haibe-Kains, Christine Desmedt, Denis Larsimont, Fatima Cardoso, Hans Peterse, Dimitry Nuyten, Marc Buyse, Marc J. Van de Vijver, Jonas Bergh, Martine Piccart, and Mauro Delorenzi. Gene expression profiling in breast cancer: Understanding the molecular basis of histologic grade to improve prognosis. *Journal of the National Cancer Institute*, 98(4):262–272, 2006.

[115] Vlad Popovici, Weijie Chen, Brandon Gallas, Christos Hatzis, Weiwei Shi, Frank Samuelson, Yuri Nikolsky, Marina Tsyganova, Alex Ishkin, Tatiana Nikolskaya, Kenneth Hess, Vicente Valero, Daniel Booser, Mauro Delorenzi, Gabriel Hortobagyi, Leming Shi, W Fraser Symmans, and Lajos Pusztai. Effect of training-sample size and classification difficulty on the accuracy of genomic predictors. *Breast Cancer Research*, 12(1):R5, 2010.

[116] Daphne R. Friedman, J. Brice Weinberg, William T. Barry, Barbara K. Goodman, Alicia D. Volkheimer, Karen M. Bond, Youwei Chen, Ning Jiang, Joseph O. Moore, Jon P. Gockerman, Louis F. Diehl, Carlos M. Decastro, Anil Potti, and Joseph R. Nevins. A genomic approach to improve prognosis and predict therapeutic response in chronic lymphocytic leukemia. *Clinical Cancer Research*, 15(22):6947–6955, 2009.

[117] Shinuk Kim, Mark Kon, and Charles DeLisi. Pathway-based classification of cancer subtypes. *Biology Direct*, 7(1):21, 2012.

[118] Zengyou He and Weichuan Yu. Stable feature selection for biomarker discovery. *Computational Biology and Chemistry*, 34(4):215 – 225, 2010.

[119] Junjie Su, Byung-Jun Yoon, and Edward Dougherty. Identification of diagnostic subnetwork markers for cancer in human protein-protein interaction network. *BMC Bioinformatics*, 11(Suppl 6):S8, 2010.

[120] Qingzhong Liu, Andrew Sung, Zhongxue Chen, Jianzhong Liu, Lei Chen, Mengyu Qiao, Zhaohui Wang, Xudong Huang, and Youping Deng. Gene selection and classification for cancer microarray data based on machine learning and similarity measures. *BMC Genomics*, 12(Suppl 5):S1, 2011.

[121] Liang Lan and Slobodan Vucetic. Multi-task feature selection in microarray data by binary integer programming. *BMC Proceedings*, 7(Suppl 7):S5, 2013.

[122] Tzu-Tsung Wong and Ching-Han Hsu. Two-stage classification methods for microarray data. *Expert Systems with Applications*, 34(1):375 – 383, 2008.

[123] Javed Khan, Jun S. Wei, Markus Ringner, Lao H. Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R. Antonescu, Carsten Peterson, and Paul S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nat Med*, 7(6):673–679, 2001.

[124] Shu-Lin Wang, Xueling Li, Shanwen Zhang, Jie Gui, and De-Shuang Huang. Tumor classification by combining pnn classifier ensemble with neighborhood rough set based gene reduction. *Comput. Biol. Med.*, 40(2):179–189, 2010.

[125] Thomas Abeel, Thibault Helleputte, Yves Van de Peer, Pierre Dupont, and Yvan Saeys. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. *Bioinformatics*, 26(3):392–398, 2010.

[126] Huawen Liu, Lei Liu, and Huijie Zhang. Ensemble gene selection by grouping for microarray data classification. *Journal of Biomedical Informatics*, 43(1): 81 – 87, 2010.

[127] Kyung-Joong Kim and Sung-Bae Cho. Ensemble classifiers based on correlation analysis for {DNA} microarray classification. *Neurocomputing*, 70 (1–3):187 – 199, 2006.

[128] Loris Nanni, Sheryl Brahnam, and Alessandra Lumini. Combining multiple approaches for gene microarray classification. *Bioinformatics*, 28(8):1151–1157, 2012.

[129] Yongjun Piao, Minghao Piao, Kiejung Park, and Keun Ho Ryu. An ensemble correlation-based gene selection algorithm for cancer classification with gene expression data. *Bioinformatics*, 28(24):3306–3315, 2012.

[130] Albert-Laszlo Barabasi, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature Reviews: Genetics*, 12(1):56–68, 2011.

[131] Kasper Lage, E. O. Karlberg, M. Størling, Zenia, Páll Í Ólason, Anders G. Pedersen, Olga Rigina, Anders M. Hinsby, Zeynep Tümer, Flemming Pociot,

Niels Tommerup, Yves Moreau, and Sø Brunak. A human phenome-interactome network of protein complexes implicated in genetic disorders. *Nature biotechnology*, 25(3):309–16, 2007.

[132] Han-Yu Chuang, Laura Rassenti, Michelle Salcedo, Kate Licon, Alexander Kohlmann, Torsten Haferlach, Robin Foà, Trey Ideker, and Thomas J. Kipps. Subnetwork-based analysis of chronic lymphocytic leukemia identifies pathways that associate with disease progression. *Blood*, 120(13):2639–2649, 2012.

[133] Ian W. Taylor, Rune Linding, David Warde-farley, Yongmei Liu, Catia Pesquita, Daniel Faria, Shelley Bull, Tony Pawson, Quaid Morris, and Jeffrey L. Wrana. Dynamic modularity in protein interaction networks predicts breast cancer outcome. *Nature biotechnology*, 27(2):199–204, 2009.

[134] Franck Rapaport, Andrei Zinovyev, Marie Dutreix, Emmanuel Barillot, and Jean-Philippe Vert. Classification of microarray data using gene networks. *BMC Bioinformatics*, 8(1):35, 2007.

[135] Igor Ulitsky, RichardM. Karp, and Ron Shamir. Detecting disease-specific dysregulated pathways via analysis of clinical expression profiles. In Martin Vingron and Limsoon Wong, editors, *Research in Computational Molecular Biology*, volume 4955 of *Lecture Notes in Computer Science*, pages 347–359. Springer Berlin Heidelberg, 2008.

[136] Michael P. Cary, Gary D. Bader, and Chris Sander. Pathway information for systems biology. {*FEBS*} *Letters*, 579(8):1815 – 1820, 2005.

[137] Silpa Suthram, Tomer Shlomi, Eytan Ruppin, Roded Sharan, and Trey Ideker. A direct comparison of protein interaction confidence assignment schemes. *BMC Bioinformatics*, 7(1):360, 2006.

[138] Minoru Kanehisa, Susumu Goto, Shuichi Kawashima, Yasushi Okuno, and Masahiro Hattori. The kegg resource for deciphering the genome. *Nucleic acids research*, 32(Database issue):D277—80, January 2004.

[139] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. M. Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, and Andrew Kasarskis. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–9, 2000.

[140] G. Joshi-Tope, M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G.R. Gopinath, G.R. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Research*, 33(suppl 1):D428–D432, 2005.

[141] Aravind Subramanian, Pablo Tamayo, Vamsi K. Mootha, Sayan Mukherjee, Benjamin L. Ebert, Michael A. Gillette, Amanda Paulovich, Scott L. Pomeroy, Todd R. Golub, Eric S. Lander, and Jill P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

[142] Da Huang, Brad Sherman, Qina Tan, Jack Collins, W Gregory Alvord, Jean Roayaei, Robert Stephens, Michael Baseler, H Clifford Lane, and Richard Lempicki. The david gene functional classification tool: a novel biological module-centric algorithm to functionally analyze large gene lists. *Genome Biology*, 8(9):R183, 2007.

[143] Enrico Glaab and Reinhard Schneider. Pathvar: analysis of gene and protein expression variance in cellular pathways using microarray data. *Bioinformatics*, 28(3):446–447, 2012.

[144] Sonja Hanzelmann, Robert Castelo, and Justin Guinney. Gsva: gene set variation analysis for microarray and rna-seq data. *BMC Bioinformatics*, 14 (1):7, 2013.

[145] Zheng Guo, Tianwen Zhang, Xia Li, Qi Wang, Jianzhen Xu, Hui Yu, Jing Zhu, Haiyun Wang, Chenguang Wang, Eric Topol, Qing Wang, and Shaoqi Rao. Towards precise classification of cancers based on robust gene functional expression profiles. *BMC Bioinformatics*, 6(1):58, 2005.

[146] Andrea H. Bild, Guang Yao, Jeffrey T. Chang, Quanli Wang, Anil Potti, Dawn Chasse, Mary-Beth Joshi, David Harpole, Johnathan M. Lancaster, Andrew Berchuck, John A. Olson, Jeffrey R. Marks, Holly K. Dressman, Mike West, and Joseph R. Nevins. Oncogenic pathway signatures in human cancers as a guide to targeted therapies. *Nature*, 439(7074):353–7, 2006.

[147] Ruoting Yang, Bernie Daigle, Linda Petzold, and Francis Doyle. Core module biomarker identification with network exploration for breast cancer metastasis. *BMC Bioinformatics*, 13(1):12, 2012.

[148] Junjie Su, Byung-Jun Yoon, and Edward R. Dougherty. Accurate and reliable cancer classification based on probabilistic inference of pathway activity. *PLoS ONE*, 4, 2009.

[149] Wei Liu, Chunquan Li, Yanjun Xu, Haixiu Yang, Qianlan Yao, Junwei Han, Desi Shang, Chunlong Zhang, Fei Su, Xiaoxi Li, Yun Xiao, Fan Zhang, Meng Dai, and Xia Li. Topologically inferring risk-active pathways toward precise cancer classification by directed random walk. *Bioinformatics*, 29(17):2169–2177, 2013.

[150] Charles J. Vaske, Stephen C. Benz, J. Zachary Sanborn, Dent Earl, Christopher Szeto, Jingchun Zhu, David Haussler, and Joshua M. Stuart. Inference of patient-specific pathway activities from multi-dimensional cancer genomics data using paradigm. *Bioinformatics*, 26(12):i237–i245, 2010.

[151] Christine Staiger, Sidney Cadot, Raul Kooter, Marcus Dittrich, Tobias Müller, Gunnar W. Klau, and Lodewyk F. A. Wessels. A critical evaluation of network and pathway-based classifiers for outcome prediction in breast cancer. *PLoS ONE*, 7(4):e34796, 2012.

[152] Xi Chen, Lily Wang, and Hemant Ishwaran. An integrative pathway-based clinical-genomic model for cancer survival prediction. *Statistics and probability letters*, 80(17-18):1313—1319, September 2010.

[153] Jr. Iannarilli, F.J. and P.A. Rubin. Feature selection for multiclass discrimination via mixed-integer linear programming. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(6):779–783, 2003.

[154] Lam C. Tsoi, Sarah L. Spain, Jo Knight, Eva Ellinghaus, Philip E. Stuart, Francesca Capon, Jun Ding, Yanming Li, Trilokraj Tejasvi, Johann E. Gudjonsson, Sari Suomela, Petra Badorf, Ulrike Hüffmeier, Werner Kurrat, Wolfgang Küster, Jesús Lascorz, Rotraut Mössner, Funda Schürmeier-Horst, Markward Ständer, Heiko Traupe, Hyun M. Kang, Judith G. M. Bergboer, Martin den Heijer, C. van de Kerkhof, Peter, Patrick L. J. M. Zeeuwen, Louise Barnes, Linda E. Campbell, Caitriona Cusack, Ciara Coleman, Judith Conroy, Sean Ennis, Michael H. Allen, Oliver Fitzgerald, Phil Gallagher, Alan D. Irvine, Brian Kirby, Trevor Markham, W. H. I. McLean, Joe McPartlin, Sarah F. Rogers, Anthony W. Ryan, Agnieszka Zawirska, Ross McManus, Emiliano Giardina, Tiziana Lepre, Carlo Perricone, Gemma

Martín-Ezquerra, Ramon M. Pujol, Eva Riveira-Munoz, Annica Inerot, Åsa T. Naluai, Lotus Mallbris, Katarina Wolk, Giuseppe Novelli, Joyce Leman, Anne Barton, Richard B. Warren, Helen S. Young, Isis Ricano-Ponce, Gosia Trynka, Fawnda J. Pellett, Andrew Henschel, Marin Aurand, Bruce Bebo, Lena Samuelsson, Christian Gieger, Thomas Illig, Susanne Moebus, Karl-Heinz Jöckel, Raimund Erbel, Peter Donnelly, Leena Peltonen, Jenefer M. Blackwell, Elvira Bramon, Matthew A. Brown, Joost Schalkwijk, Juan P. Casas, Aiden Corvin, Nicholas Craddock, Audrey Duncanson, Janusz Jankowski, Hugh S. Markus, Christopher G. Mathew, Mark I. McCarthy, Colin N. A. Palmer, Robert Plomin, Mona Ståhle, Anna Rautanen, Stephen J. Sawcer, Nilesh Samani, Ananth C. Viswanathan, Nicholas W. Wood, Cé Bellenguez, Colin Freeman, Garrett Hellenthal, Eleni Giannoulatou, Matti Pirinen, A. D. Burden, Zhan Su, Sarah E. Hunt, Rhian Gwilliam, Suzannah J. Bumpstead, Serge Dronov, Matthew Gillman, Emma Gray, Naomi Hammond, Alagurevathi Jayakumar, Owen T. McCann, Catherine H. Smith, Jennifer Liddle, Marc L. Perez, Simon C. Potter, Radhi Ravindrarajah, Michelle Ricketts, Matthew Waller, Paul Weston, Sara Widaa, Pamela Whittaker, Michael J. Cork, Xavier Estivill, Anne M. Bowcock, Gerald G. Krueger, Wolfgang Weger, Jane Worthington, Rachid Tazi-Ahnini, Frank O. Nestle, Adrian Hayday, Per Hoffmann, Juliane Winkelmann, Cisca Wijmenga, Cordelia Langford, Sarah Edkins, Robert Andrews, Hannah Blackburn, Amy Strange, Gavin Band, Richard D. Pearson, Damjan Vukcevic, Chris C. A. Spencer, Panos Deloukas, Ulrich Mrowietz, Stefan Schreiber, Stephan Weidinger, Sulev Koks, Kü Kingo, Tonu Esko, Andres Metspalu, Henry W. Lim, John J. Voorhees, Michael Weichenthal, H. E. Wichmann, Vinod Chandran, Cheryl F. Rosen, Proton Rahman, Dafna D. Gladman, Christopher E. M. Griffiths, Andre Reis, Juha Kere, Rajan P. Nair, Andre Franke, Jonathan N. W. N. Barker, Goncalo R. Abecasis, James T. Elder, Richard C. Trembath, Kristina C. Duffin, Cindy Helms, David Goldgar, Yun Li, Justin Paschall, Mary J. Malloy, Clive R. Pullinger, John P. Kane, Jennifer Gardner, Amy Perlmutter, Andrew Miner, Bing J. Feng, Ravi Hiremagalore, Robert W. Ike, Enno Christophers, Tilo Henseler, Andreas Ruether, Steven J. Schrodi, Sampath Prahalad, Stephen L. Guthery, Judith Fischer, Wilson Liao, Pui Kwok, Alan Menter, G. M. Lathrop, C. Wise, Ann B. Begovich, Alexandros Onoufriadis, Michael E. Weale, Angelika Hofer, Wolfgang Salmhofer, Peter Wolf, Kati Kainu, and Ulpu Saarialho-Kere. Identification

of 15 new psoriasis susceptibility loci highlights the role of innate immunity. *Nature genetics*, 44(12):1346–8, 2012.

[155] Frank O. Nestle, Daniel H. Kaplan, and Jonathan Barker. Psoriasis. *New England Journal of Medicine*, 361(5):496–509, 2009.

[156] Saravana M. Dhanasekaran, Terrence R. Barrette, Debashis Ghosh, Rajal Shah, and et al. Delineation of prognostic biomarkers in prostate cancer. *Nature*, 412(6849):822–6, 2001.

[157] Ash A. Alizadeh, Michael B. Eisen, R. E. Davis, Chi Ma, Izidore S. Lossos, Andreas Rosenwald, Jennifer C. Boldrick, Hajeer Sabet, Truc Tran, Xin Yu, John I. Powell, Liming Yang, Gerald E. Marti, Troy Moore, James Hudson, Lisheng Lu, David B. Lewis, Robert Tibshirani, Gavin Sherlock, Wing C. Chan, Timothy C. Greiner, Dennis D. Weisenburger, James O. Armitage, Roger Warnke, Ronald Levy, Wyndham Wilson, Michael R. Grever, John C. Byrd, David Botstein, Patrick O. Brown, and Louis M. Staudt. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511, 2000.

[158] William R. Swindell, Andrew Johnston, Steve Carbajal, Gangwen Han, Christian Wohn, Jun Lu, Xianying Xing, Rajan P. Nair, John J. Voorhees, James T. Elder, Xiao-Jing Wang, Shigetoshi Sano, Errol P. Prens, John DiGiovanni, Mark R. Pittelkow, Nicole L. Ward, and Johann E. Gudjonsson. Genome-wide expression profiling of five mouse models identifies similarities and differences with human psoriasis. *PLoS ONE*, 6:e18266, 2011.

[159] P. Farmer, H. Bonnefoi, V. Becette, M. Tubiana-Hulin, P. Fumoleau, D. Larsimont, G. MacGrogan, J. Bergh, D. Cameron, D. Goldstein, S. Duss, A-L Nicoulaz, C. Brisken, and R. Iggo. Identification of molecular apocrine breast tumours by microarray analysis. *Oncogene*, 24(29):4660–4671, 2005.

[160] G.K. Smyth. limma: Linear models for microarray data. In Robert Gentleman, VincentJ. Carey, Wolfgang Huber, RafaelA. Irizarry, and Sandrine Dudoit, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, Statistics for Biology and Health, pages 397–420. Springer New York, 2005. ISBN 978-0-387-25146-2.

[161] Arthur Liberzon, Aravind Subramanian, Reid Pinchback, Helga Thorvaldsdóttir, Pablo Tamayo, and Jill P. Mesirov. Molecular signatures database (msigdb) 3.0. *Bioinformatics*, 27(12):1739–1740, 2011.

[162] Louise R. Howe and Anthony M.C. Brown. Wnt signaling and breast cancer. *Cancer Biology & Therapy*, 3(1):36–41, 2004.

[163] Rebecca Lamb, Matthew P. Ablett, Katherine Spence, Göran Landberg, Andrew H. Sims, and Robert B. Clarke. Wnt pathway activity in breast cancer sub-types and stem-like cells. *PLoS ONE*, 8:e67811, 2013.

[164] Devina Mitra, Stephan Bernhardt, Zita Soons, Gernot Poschet, Ruediger Hell, Rainer Koenig, Ulrike Korf, and Stefan Wiemann. Metabolic transformations in breast cancer subtypes. *Cancer and Metabolism*, 2(Suppl 1):P4, 2014.

[165] Tomohiko Ohta and Mamoru Fukuda. Ubiquitin and breast cancer. *Oncogene*, 23(11):2079–88, 2004.

[166] Diana Cepeda, Hwee-Fang Ng, Hamid Reza Sharifi, Salah Mahmoudi, Vanessa Soto Cerrato, Erik Fredlund, Kristina Magnusson, Helén Nilsson, Alena Malyukova, Juha Rantala, Daniel Klevebring, Francesc Viñals, Nimesh Bhaskaran, Siti Mariam Zakaria, Aldwin Suryo Rahmanto, Stefan Grotegut, Michael Lund Nielsen, Cristina Al-Khalili Szigyarto, Dahui Sun, Mikael Lerner, Sanjay Navani, Martin Widschwendter, Mathias Uhlén, Karin Jirström, Fredrik Pontén, James Wohlschlegel, Dan Grandér, Charles Spruck, Lars-Gunnar Larsson, and Olle Sangfelt. Cdk-mediated activation of the scffbxo28 ubiquitin ligase promotes myc-driven transcription and tumourigenesis and predicts poor survival in breast cancer. *EMBO Molecular Medicine*, 5(7):1067–1086, 2013.

[167] J Taylor-Papadimitriou, J M Burchell, D Miles, and R Sewell. Changes in mucin-type o-glycosylation in breast cancer: implications for the host immune response. *International Journal of Experimental Pathology*, 85(4): A52 – A52, 2004.

[168] Danni Meany and Daniel Chan. Aberrant glycosylation associated with enzymes as cancer biomarkers. *Clinical Proteomics*, 8(1):7, 2011.

[169] Seema Iyer, Deana M. Ferreri, Nina C. DeCocco, Fred L. Minnear, and Peter A. Vincent. Ve-cadherin-p120 interaction is required for maintenance of

endothelial barrier function. *American Journal of Physiology - Lung Cellular and Molecular Physiology*, 286(6):L1143–L1153, 2004.

[170] Mehran Haidari, Wei Zhang, and Koji Wakame. Disruption of endothelial adherens junction by invasive breast cancer cells is mediated by reactive oxygen species and is attenuated by {AHCC}. *Life Sciences*, 93(25–26):994 – 1003, 2013.

[171] Yun Bai, Pu Wang, Chuan Li, Jingjing Xie, and Yin Wang. A multi-scale relevance vector regression approach for daily urban water demand forecasting. *Journal of Hydrology*, 517(0):236 – 245, 2014.

[172] Kamini Venkatesh, Vadlamani Ravi, Anita Prinzie, and Dirk Van den Poel. Cash demand forecasting in atms by clustering and neural networks. *European Journal of Operational Research*, 232(2):383 – 392, 2014.

[173] Paulo Cortez, António Cerdeira, Fernando Almeida, and Telmo Matos a2009nd José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547 – 553, 2009.

[174] Eddie Davis and Marianthi Ierapetritou. A kriging-based approach to minlp containing black-box models and noise. *Industrial and Engineering Chemistry Research*, 47(16):6101–6125, 2008.

[175] José A. Caballero and Ignacio E. Grossmann. An algorithm for the use of surrogate models in modular flowsheet optimization. *AIChE Journal*, 54 (10):2633–2650, 2008.

[176] Carlos A. Henao and Christos T. Maravelias. Surrogate-based superstructure optimization framework. *AIChE Journal*, 57(5):1216–1232, 2011.

[177] Carlos A. Henao and Christos T. Maravelias. Surrogate-based process synthesis. In S. Pierucci and G. Buzzi Ferraris, editors, *20th European Symposium on Computer Aided Process Engineering*, volume 28 of *Computer Aided Chemical Engineering*, pages 1129 – 1134. Elsevier, 2010.

[178] Felipe A. C. Viana, Timothy W. Simpson, Vladimir Balabanov, and Vasilli Toropov. Metamodeling in Multidisciplinary Design Optimization: How Far Have We Really Come? *AIAA Journal*, 52(4):670–690, 2014.

[179] Joakim Beck, Daniel Friedrich, Stefano Brandani, Serge Guillas, and Eric S Fraga. Surrogate based optimisation for design of pressure swing adsorption systems. In *Proceedings of the 22nd European Symposium on Computer Aided Process Engineering*, 2012.

[180] Haralambos Sarimveis, Alex Alexandridis, Stefanos Mazarakis, and George Bafas. A new algorithm for developing dynamic radial basis function neural network models based on genetic algorithms. *Computers and Chemical Engineering*, 28(1–2):209 – 217, 2004.

[181] André I. Khuri and Siuli Mukhopadhyay. Response surface methodology. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(2):128–149, 2010.

[182] M. Khayet, C. Cojocaru, and G. Zakrzewska-Trznadel. Response surface modelling and optimization in pervaporation. *Journal of Membrane Science*, 321(2):272 – 283, 2008.

[183] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[184] Robert Tibshirani. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(3):273–282, 2011.

[185] Alison Cozad, Nikolaos V. Sahinidis, and David C. Miller. Learning surrogate models for simulation-based optimization. *AIChE Journal*, 60(6): 2211–2227, 2014.

[186] Y. Zhang and N. V. Sahinidis. Uncertainty quantification in co2 sequestration using surrogate models from polynomial chaos expansion. *Industrial and Engineering Chemistry Research*, 52(9):3121–3132, 2013.

[187] David C. Miller, Madhava Syamlal, David S. Mebane, Curt Storlie, Debangsu Bhattacharyya, Nikolaos V. Sahinidis, Deb Agarwal, Charles Tong, Stephen E. Zitney, Avik Sarkar, Xin Sun, Sankaran Sundaresan, Emily Ryan, Dave Engel, and Crystal Dale. Carbon capture simulation initiative: A case study in multiscale modeling and new challenges. *Annual Review of Chemical and Biomolecular Engineering*, 5(1):301–323, 2014. PMID: 24797817.

[188] AlexJ. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[189] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3): 27:1–27:27, 2011.

[190] Chi-Jie Lu, Tian-Shyug Lee, and Chih-Chou Chiu. Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems*, 47(2):115 – 125, 2009.

[191] Vladimir Cherkassky and Yunqian Ma. Practical selection of svm parameters and noise estimation for svm regression. *Neural Networks*, 17(1):113 – 126, 2004.

[192] J. P. C. Kleijnen and W. C. M. van Beers. Application-driven sequential designs for simulation experiments: Kriging metamodelling. *The Journal of the Operational Research Society*, 55(8):pp. 876–883, 2004.

[193] Jack P.C. Kleijnen. Kriging metamodeling in simulation: A review. *European Journal of Operational Research*, 192(3):707 – 716, 2009.

[194] C. D. Lloyd and P. M. Atkinson. Deriving dsms from lidar data with kriging. *International Journal of Remote Sensing*, 23(12):2519–2524, 2002.

[195] Dick J. Brus and Gerard B.M. Heuvelink. Optimization of sample patterns for universal kriging of environmental variables. *Geoderma*, 138(1–2):86 – 95, 2007.

[196] Michael. S. Balshi, A. David Mcguire, Paul Duffy, Mike Flannigan, John Walsh, and Jerry Melillo. Assessing the response of area burned to changing climate in western boreal north america using a multivariate adaptive regression splines (mars) approach. *Global Change Biology*, 15(3):578–600, 2009.

[197] Tim Hill, Leorey Marquez, Marcus O'Connor, and William Remus. Artificial neural network models for forecasting and decision making. *International Journal of Forecasting*, 10(1):5 – 15, 1994.

[198] Muriel Gevrey, Ioannis Dimopoulos, and Sovan Lek. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological Modelling*, 160(3):249 – 264, 2003.

[199] Mukta Paliwal and Usha A. Kumar. Neural networks and statistical techniques: A review of applications. *Expert Systems with Applications*, 36(1):2 – 17, 2009.

[200] V.G. Gudise and G.K. Venayagamoorthy. Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks. In *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pages 110–117, 2003.

[201] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[202] J. R. Quinlan. Learning with continuous classes. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 343–348. World Scientific, 1992.

[203] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[204] Gérard Biau. Analysis of a random forests model. *Journal of Machine Learning Research*, 13:1063–1095, 2012.

[205] Robin Genuer, Jean-Michel Poggi, and Christine Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225 – 2236, 2010.

[206] G. Fanelli, J. Gall, and L. Van Gool. Real time head pose estimation with random regression forests. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 617–624, 2011.

[207] Kari T. Korhonen and Annika Kangas. Application of nearest neighbour regression for generalizing sample tree information. *Scandinavian Journal of Forest Research*, 12(1):97–101, 1997.

[208] Harris Papadopoulos, Vladimir Vovk, and Alex Gammerman. Regression conformal prediction with nearest neighbours. *Journal of Artificial Intelligence Research*, 40(1):815–840, 2011.

[209] Judith D. Toms and Mary L. Lesperance. Piecewise regression: A tool for identifying ecological thresholds. *Ecology*, 84(8):2034–2041, 2003.

[210] Birgit Strikholm. Determining the number of breaks in a piecewise linear regression model. Working Paper Series in Economics and Finance 648, Stockholm School of Economics, 2006.

[211] Gihan F. Malash and Mohammad I. El-Khaiary. Piecewise linear regression: A statistical method for the analysis of experimental adsorption data by the intraparticle-diffusion models. *Chemical Engineering Journal*, 163(3):256 – 263, 2010.

[212] Vito MR Muggeo. Segmented: an r package to fit regression models with broken-line relationships. *R news*, 8(1):20–25, 2008.

[213] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008.

[214] Vito M. R. Muggeo. Estimating regression models with unknown breakpoints. *Statistics in Medicine*, 22(19):3055–3071, 2003.

[215] Alessandro Magnani and StephenP. Boyd. Convex piecewise-linear fitting. *Optimization and Engineering*, 10(1):1–17, 2009.

[216] Alejandro Toriello and Juan Pablo Vielma. Fitting piecewise linear continuous functions. *European Journal of Operational Research*, 219(1):86–95, 2012.

[217] K. Palmer and M. Realff. Metamodeling approach to optimization of steady-state flowsheet simulations: Model generation. *Chemical Engineering Research and Design*, 80(7):760 – 772, 2002.

[218] J.C. Helton and F.J. Davis. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering and System Safety*, 81(1):23 – 69, 2003.

[219] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[220] Athanasios Tsanas and Angeliki Xifara. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. *Energy and Buildings*, 49(0):560 – 567, 2012.

[221] I.-C. Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12):1797 – 1808, 1998.

[222] Gints Jekabsons. Areslab: Adaptive regression splines toolbox for matlab/octave, 2011.

[223] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, GeoffreyJ. McLachlan, Angus Ng, Bing Liu, PhilipS. Yu, Zhi-Hua Zhou, Michael Steinbach, DavidJ. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1): 1–37, 2008.

[224] Leo Breiman. Statistical modeling: The two cultures. *Statistical Science*, 2001.

[225] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[226] Ross J. Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, Singapore, 1992. World Scientific.

[227] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. In *Poster papers of the 9th European Conference on Machine Learning*. Springer, 1997.

[228] Max Kuhn, Steve Weston, Chris Keefer, and Nathan Coulter. Cubist models for regression, 2012.

[229] Amit Thombre. Comparing logistic regression, neural networks, c5.0 and m5 classification techniques. In *Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'12, pages 132–140, Berlin, Heidelberg, 2012. Springer-Verlag.

[230] Probal Chaudhuri, Min ching Huang, Wei yin Loh, and Ruji Yao. Piecewise-polynomial regression trees. *Statistica Sinica*, 4:143–167, 1994.

[231] Wei Y. Loh. Regression Trees with Unbiased Variable Selection and Interaction Detection. *Statistica Sinica*, 12(2):361–386, 2002.

[232] Celine Vens and Hendrik Blockeel. A simple regression based heuristic for learning model trees. *Intell. Data Anal.*, 10(3):215–236, 2006.

[233] Donato Malerba, Floriana Esposito, Michelangelo Ceci, and Annalisa Appice. Top-down induction of model trees with regression and splitting nodes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):612–625, 2004.

[234] Amos Hong and Argon Chen. Piecewise regression model construction with sample efficient regression tree (sert) and applications to semiconductor yield analysis. *Journal of Process Control*, 22(7):1307 – 1317, 2012.

[235] Thomas Grubinger, Achim Zeileis, and Karl-Peter Pfeiffer. evtree: Evolutionary learning of globally optimal classification and regression trees in r. *Journal of Statistical Software*, 61(1), 2014.

[236] Zhiwei Fu, Bruce L. Golden, Shreevardhan Lele, S. Raghavan, and Edward A. Wasil. A genetic algorithm-based approach for building accurate decision trees. *INFORMS Journal on Computing*, 15(1):3–22, 2003.

[237] R.C. Barros, M.P. Basgalupp, A.C.P.L.F. de Carvalho, and A.A. Freitas. A survey of evolutionary algorithms for decision-tree induction. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(3):291–312, 2012.

[238] Pu Wang, Ke Tang, E.P.K. Tsang, and Xin Yao. A memetic genetic programming with decision tree-based local search for classification problems. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 917–924, 2011.

[239] Terry M. Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning*, 2011.

[240] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486 (3–5):75 – 174, 2010.

[241] Alexander W. Rives and Timothy Galitski. Modular organization of cellular networks. *Proceedings of the National Academy of Sciences*, 100(3):1128–1133, 2003.

[242] Natali Gulbahce and Sune Lehmann. The art of community detection. *BioEssays*, 30(10):934–938, 2008.

[243] Yon Dourisboure, Filippo Geraci, and Marco Pellegrini. Extraction and classification of dense communities in the web. In *Proceedings of the 16th*

International Conference on World Wide Web, WWW '07, pages 461–470, New York, NY, USA, 2007. ACM.

[244] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20(2):172–188, 2008.

[245] Leon Danon, Albert Díaz-guilera, and Jordi Duch. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, page 09008, 2005.

[246] C.O. Dorso A. Medus, G. Acuña. Detection of community structures in networks via global optimization. *Physica A: Statistical Mechanics and its Applications*, 358(2–4):593 – 604, 2005.

[247] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72:027104, 2005.

[248] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[249] Y. Liu, J. Moser, and S. Aviyente. Network community structure detection for directional neural networks inferred from multichannel multisubject eeg data. *IEEE Transactions on Biomedical Engineering*, 61(7):1919–1930, 2014.

[250] Darong Lai, Hongtao Lu, and Christine Nardini. Extracting weights from edge directions to find communities in directed networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2010(06):P06003, 2010.

[251] Darong Lai, Hongtao Lu, and Christine Nardini. Finding communities in directed networks by pagerank random walk induced network embedding. *Physica A: Statistical Mechanics and its Applications*, 389(12):2443 – 2454, 2010.

[252] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.

[253] C. Wohn, A. Brand, K. van Ettinger, I. Brouwers-Haspels, A. Waisman, J.D. Laman, and Björn E. Clausen. Gradual development of psoriatic skin lesions by constitutive low-level expression of il-17a. *Cellular Immunology*, pages –, 2015.

[254] Dmitry Svetlichnyy, Hana Imrichova, Mark Fiers, Zeynep Kalender Atak, and Stein Aerts. Identification of high-impact ¡italic¿cis¡/italic¿-regulatory mutations using transcription factor specific random forest models. *PLoS Comput Biol*, 11(11):e1004590, 11 2015.

[255] Dorothea Terhorst, Rabie Chelbi, Christian Wohn, Camille Malosse, Samira Tamoutounour, Audrey Jorquera, Marc Bajenoff, Marc Dalod, Bernard Malissen, and Sandrine Henri. Dynamics and transcriptomics of skin dendritic cells and macrophages in an imiquimod-induced, biphasic mouse model of psoriasis. *The Journal of Immunology*, 195(10):4953–4961, 2015.

[256] James C. Chen, Jane E. Cerise, Ali Jabbari, Raphael Clynes, and Angela M. Christiano. Master regulators of infiltrate recruitment in autoimmune disease identified through network-based molecular deconvolution. *Cell Systems*, 1 (5):326 – 337, 2015.

[257] M.H. Wang, K. Tsoi, Xin Lai, M. Chong, B. Zee, Tian Zheng, Shaw-Hwa Lo, and Inchi Hu. Two screening methods for genetic association study with application to psoriasis microarray data sets. In *Big Data (BigData Congress), 2015 IEEE International Congress on*, pages 324–326, 2015.

[258] Sudharsana Sundarrajan, Sajitha Lulu, and Mohanapriya Arumugam. Insights into protein interaction networks reveal non-receptor kinases as significant druggable targets for psoriasis. *Gene*, 566(2):138 – 147, 2015.

[259] Pascal Jézéquel, Zein Sharif, Hamza Lasla, Wilfried Gouraud, Catherine Guérin-Charbonnel, Loïc Campion, Stéphane Chrétien, and Mario Campone. Gene-expression signature functional annotation of breast cancer tumours in function of age. *BMC Medical Genomics*, 8(1):80, 2015.

[260] Hugo Tovar, Rodrigo García-Herrera, Jesús Espinal-Enríquez, and Enrique Hernández-Lemus. Transcriptional master regulator analysis in breast cancer genetic networks. *Computational Biology and Chemistry*, 59, Part B:67 – 77, 2015.

[261] MohammedA. Aleskandarany, IanO. Ellis, and EmadA. Rakha. Molecular classification of breast cancer. In Ashraf Khan, Ian O. Ellis, Andrew M. Hanby, Ediz F. Cosar, Emad A. Rakha, and Dina Kandil, editors, *Precision Molecular Pathology of Breast Cancer*, volume 10 of *Molecular Pathology Library*, pages 137–155. Springer New York, 2015.

[262] Yi Shen, Zhanwei Wang, LenoraWM Loo, Yan Ni, Wei Jia, Peiwen Fei, HarveyA. Risch, Dionyssios Katsaros, and Herbert Yu. Linc00472 expression is regulated by promoter methylation and associated with disease-free survival in patients with grade 2 breast cancer. *Breast Cancer Research and Treatment*, 154(3):473–482, 2015.

[263] D.J. Dittman, T.M. Khoshgoftaar, and A. Napolitano. The effect of data sampling when using random forest on imbalanced bioinformatics data. In *Information Reuse and Integration (IRI), 2015 IEEE International Conference on*, pages 457–463, 2015.

[264] B. Heredia, T.M. Khoshgoftaar, A. Fazelpour, and D.J. Dittman. Building an effective classification model for breast cancer patient response data. In *Information Reuse and Integration (IRI), 2015 IEEE International Conference on*, pages 229–235, 2015.

[265] Xiaoming Wang and Shitong Wang. Enhanced algorithm for high-dimensional data classification. *Applied Soft Computing*, 40:1 – 9, 2016.

[266] Lijun Han, Changjun Zhou, Bin Wang, and Qiang Zhang. A combining dimensionality reduction approach for cancer classification. In Antonis Bikakis and Xianghan Zheng, editors, *Multi-disciplinary Trends in Artificial Intelligence*, volume 9426 of *Lecture Notes in Computer Science*, pages 340–347. Springer International Publishing, 2015.

[267] Khalid Benabdeslem, Haytham Elghazel, and Mohammed Hindawi. Ensemble constrained laplacian score for efficient and robust semi-supervised feature selection. *Knowledge and Information Systems*, pages 1–25, 2015.

[268] Rok Blagus and Lara Lusa. Joint use of over- and under-sampling techniques and cross-validation for the development and assessment of prediction models. *BMC Bioinformatics*, 16(1):362, 2015.

[269] Zakariya Yahya Algamal and Muhammad Hisyam Lee. Regularized logistic regression with adjusted adaptive elastic net for gene selection in high dimensional cancer classification. *Computers in Biology and Medicine*, 67:136 – 145, 2015.

[270] Afef Ben Brahim and Mohamed Limam. A hybrid feature selection method based on instance learning and cooperative subset search. *Pattern Recognition Letters*, 69:28 – 34, 2016.

[271] Tong Wu, Xudong Wang, Jing Li, Xiuzhen Song, Ying Wang, Yunfeng Wang, Lei Zhang, Ziyao Li, and Jiawei Tian. Identification of personalized chemoresistance genes in subtypes of basal-like breast cancer based on functional differences using pathway analysis. *PLoS ONE*, 10(6):e0131183, 06 2015.

[272] Brian Lehmann, Yan Ding, Daniel Viox, Ming Jiang, Yi Zheng, Wang Liao, Xi Chen, Wei Xiang, and Yajun Yi. Evaluation of public cancer datasets and signatures identifies tp53 mutant signatures with robust prognostic and predictive value. *BMC Cancer*, 15(1):179, 2015. Brian David Lehmann and Yan Ding contributed equally to this work.

[273] Alan Mackay, Britta Weigelt, Anita Grigoriadis, Bas Kreike, Rachael Natrajan, Roger A'Hern, David S.P. Tan, Mitch Dowsett, Alan Ashworth, and Jorge S. Reis-Filho. Microarray-based class discovery for molecular classification of breast cancer: Analysis of interobserver agreement. *Journal of the National Cancer Institute*, 103(8):662–673, 2011.

[274] Tara Sanft, Bilge Aktas, Brock Schroeder, Veerle Bossuyt, Michael DiGiovanna, Maysa Abu-Khalaf, Gina Chung, Andrea Silber, Erin Hofstatter, Sarah Mougalian, Lianne Epstein, Christos Hatzis, Cathy Schnabel, and Lajos Pusztai. Prospective assessment of the decision-making impact of the breast cancer index in recommending extended adjuvant endocrine therapy for patients with early-stage er-positive breast cancer. *Breast Cancer Research and Treatment*, 154(3):533–541, 2015.

[275] Cagatay Arslan, M.Kadri Altundag, and Y.Yavuz Ozisik. Gene arrays, prognosis, and therapeutic interventions. In Adnan Aydiner, Abdullah İğci, and Atilla Soran, editors, *Breast Disease*, pages 207–227. 2016.

[276] JuanCruz Rodriguez, Germán González, Cristobal Fresno, and Elmer A. Fernández. Integrative functional analysis improves information retrieval in breast cancer. In Alvaro Pardo and Josef Kittler, editors, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, volume 9423 of *Lecture Notes in Computer Science*, pages 43–50. Springer International Publishing, 2015.