Multi-criteria optimisation for complex learning prediction systems

BASSMA AL-JUBOURI

A thesis submitted in partial fulfilment of the requirements of
Bournemouth University for the degree of

Doctor of Philosophy

First Supervisor: Prof. Bogdan Gabrys
Second Supervisor: Dr. Emili Balaguer-Ballester

May 2018

Copyright statement

Abstract

This work presents a framework for the inclusion of multiple criteria in the design process of supervised learning algorithms; as well as studies the sophisticated interactions among them. The criteria included and tested experimentally in this thesis are: accuracy, model complexity, algorithmic complexity, diversity and robustness.

The present thesis addresses important challenges related to considering multiple criteria such as: 1) defining suitable measures for the included criteria, 2) determining effective approaches to optimise the system performance using multiple objectives, 3) finding effective alternative approaches to include such criteria indirectly in the design stages when defining accurate measures is infeasible, and finally 4) analysing the possible interactions among the criteria as well as identifying the main factors/decision points that modulate them.

This work introduces a novel Multi-Components, Multi-Layer Predictive System (MCMLPS). This system incorporates mechanisms designed to control the diversity, model complexity and robustness.

In the first stage of this thesis, the accuracy, model and algorithmic complexities of the base components for the proposed system have been optimised empirically using two multi-objective optimisation approaches. The first approach consists of a scalarized multi-objective optimisation, where the models are generated from optimising a single cost function that combines the three criteria. The second approach uses a Pareto-based multi objective optimisation which establishes a trade-off among the three criteria to generate a set of selectively balanced models.

These first results showed that models generated from Pareto-based multi objective optimisation approach are both more accurate and more diverse than the models generated from scalarized multi-objective optimisation approach. However, the Pareto-based approach is hindered by the high algorithmic complexity required to find the best model and the infeasibility of defining universal measures for some of the above-mentioned criteria. Thus, in later stages of this work these criteria are either presented as constraints or included indirectly in generating the base components for the MCMLPS.

In a subsequent stage of this study, the diversity among the base components of the proposed MCMLPS system is encouraged by training them on local regions in the data, were the locality is determined using the similarity of the data features. Each local region contains either disjoint subsets of the data and/or subsets of the features. A range of similarity metrics such as pairwise squared correlation and conditional mutual informa-

tion of the features are used. Interestingly, the squared correlation method can be applied in supervised as well as unsupervised learning as it does not consider the output class when splitting the data. Meanwhile, the conditional mutual information method can be applied only in supervised learning as it uses the output class in splitting the data. The full MCMLPS architecture is then analysed and its performance is compared to three well-known ensemble methods.

Next, the effect of weighing the components of the MCMLPS and combining them is examined using six fusion methods. The results showed that, including the similarity metric used to divide the data into local regions in weighing the system components, often results in the best accuracy compared to the other fusion methods.

In the final phase of this study, the robustness of the proposed system in noisy environments is tested and compared to other ensemble methods. The system showed a comparable accuracy to the best performing ensemble and it often has a more robust performance than other ensembles in highly noisy environments.

To conclude, the present thesis proposes a multi-component, multi-layer system which simultaneously incorporates multiple criteria in its design cycle. The results of this thesis suggest that the locality in learning and high diversity among the components of the proposed system can be particularly beneficial in designing ensemble learning methods for highly noisy data sets.

Contents

List of Figures

List of Tables

Nomenclature

$\mathbf{X}$	Matrix
$\mathbf{x}$	Vector
x	Scalar value
$\mathcal{X}$	Domain
$\mathcal{D}$	Dataset
$\hat{\mathbf{y}}$	Predicted value
$P(x)$	Probability function
$L(x)$	Loss function
$Cr(x)$	Criterion function
$C(i, j)$	Cost function of prediction
$\Omega()$	Model complexity
$O()$	Big O notation
λ	Regularization hyperparameter
$\mathbf{W}$	Weight matrix
$I(x_1, x_2)$	Mutual information for x_1 and x_2
$RLA_{x\%noise}$	Relative Loss of Accuracy when $x\%$ noise is added to the data

List of Abbreviations

SVM	Support Vector Machine
MCMLPS	Multi-Components, Multi-Layer, Predictive System
MOO	Multi-Objective Optimization
MSE	Mean Square Error
DT	Decision Trees
NN	Neural Network
NSGA	Non-dominated Sorting Genetic Algorithm
SOO	Single Objective Optimisation
MLP	Multi-Layer Perceptron
PCA	Principle Component Analysis
LDA	Linear Discriminate Analysis
ICA	Independent Component Analysis
MI	Mutual Information
LR	Local Region
DPS	Density Preserving Sampling
ZCA	Zero-phase Component Analysis
CV	Cross Validation
RF	Rotation Forest
CMI	Conditional Mutual Information
WMV	Weighted Majority Vote
RLA	Relative Loss of Accuracy
MV	Majority Vote

Acknowledgements

First of all, I would like to express my gratitude to my supervisor Prof. Bogdan Gabrys for his guidance and support throughout this project. I am thankful for the time he spent discussing this work, and for his expert knowledge and invaluable feedback which helped me in developing and improving many aspects of this work.
My sincere thanks also go to my second supervisor Dr. Emili Balaguer-Ballester for his constant encouragement and support during my PhD. I am grateful for his constructive feedback and quick response which he had provided no matter how busy he was.
I am thankful to Dr. Marcin Budka and Prof. Hamid Bouchachia for their constructive feedback during the transfer. My thanks to Dr. Marcin for the comments and insights I had received from him during the writing of this thesis which helped me to improve the presentation of this work. Also, I am thankful to my former supervisor Dr. Jianbing Ma for his input in the early stages of this project. Others who contributed to this work are the anonymous reviewers whom their valuable feedback had helped me in improving and strengthening the work presented in this thesis.
Also, my thanks go to Naomi Bailey for her administrative advice and support during my PhD. Many thanks for my colleagues and friends in Bournemouth and back home for their help, support and encouragement.
I am deeply grateful to my family for their unconditional love and support, especially my parents who raised me with a love for science and support me in all my pursuits. Finally, I am thankful to the bless in my life, my sister, my colleague and my best friend Abeer. I am grateful for the long hours she spent discussing and reviewing the work presented in this thesis with me and for keeping me motivated at all time. Without her encouragement, love and support I would not be able to undertake this project.

Declaration

The work contained in this thesis is the result of my own investigations and has not been accepted nor concurrently submitted in candidature for any other award.

To my parents...

Mona and Khaldoon

Chapter 1

Introduction

Machine learning is considered as a relatively new field of research, however, collecting data and recognizing distinctive patterns from it can be traced back at least to the 16th century, when Tycho Brahe recorded his astronomical observations (Dreyer (1890) and Swerdlow (1996)); which enabled Johannes Kepler to discover empirical laws of planetary motion (Small (1804) and Bishop (1995)). While pattern recognition has a long standing history, however, by the end of the first half of the 20th century all the main results in this field stem from statistics (Theodoridis and Koutroumbas (2006)). In the second half of the 20th century, machine learning had witnessed important developments, starting from the work of Alan Turning in the 1950s (Turing (1950)) and continuing with the introduction of the first Neural Network (NN) machine by Marvin Minsky and Dean Edmonds (Russell and Norvig (2010)), and later with the development of the perceptron (Rosenblatt (1958)). During the AI winter in 1970s, the intensity of the research in this field slowed down (Cervier (1993)); but shortly thereafter, the exponential increase in computer power fostered again machine learning research (Theodoridis and Koutroumbas (2006)). Highly sophisticated algorithms such as deep learning or ensembles of learners that were infeasible in terms of their computational cost few decades ago are now commonly used in complex real-life settings (Deng and Yu (2014) and Zhang and Ma (2012)). Historically, machine learning systems have been designed to optimize the generalization ability of the system as the main criterion. The traditional aim of machine learning is to design a system that can effectively learn regularities in the training data and then uses these identified regularities to perform tasks in the future, such as classifying the data into different classes with optimal accuracy (Bishop (1995)).

Nevertheless, as new learning systems which tackle different types of problems were introduced over the years, the need to include other criteria in the optimisation process of machine learning systems emerged (Jin (2006)). The concept of including more than one criterion in optimising the performance of machine learning systems has been the focus on a lively debate in the literature over the last two decades, for example:

- The inclusion of accuracy and model complexity has been revisited from the statistical learning (Vapnik (2013)) and Bayesian angles (Močkus (1975)): the developed systems should be accurate, yet their complexities should be bounded so that a complex system is not chosen for a problem that can be solved using a simpler system (Blumer et al. (1987)).
- The inclusion of accuracy and diversity: when multiple models are combined to produce the system, the diversity among them should be encouraged and monitored during the design stages (Cunningham and Carney (2000)). An ensemble with diverse models can have better performance due to the complementary behaviour of its components (Xue et al. (2006)).
- The inclusion of accuracy and robustness: the developed system should be able to tolerate certain variation in the data, yet it should be robust to outliers and noise (Xu et al. (2009)).

The above points shows that, previous approaches typically focus on pairs of criteria to optimise, in which one of them is the accuracy of the prediction (Jin (2006)). However, the interaction among these criteria and the effect they have on each other is rarely discussed. The aim of this work is to design a novel framework which accounts for the multiple criteria in the optimization process for complex machine learning systems; and to study the effect of considering these criteria in the system performance in noisy datasets. Specifically, the criteria which are investigated in this work are the accuracy, model complexity, algorithmic complexity, diversity and robustness; which provide a rich description for the ensemble learning process from multiple angles. Furthermore, this work studies the feasibility of measuring these criteria, the advantages and drawbacks of including them in the optimization process, and most importantly the possible interactions among them.

1.1 Background

Predictive models that are built and trained with an overreliance on the prediction accuracy as the main optimisation criterion can result, as it is well-known, in some form of overfitting. Thus, heavily penalizing the complexity of the model whilst maintaining its accuracy has been a common strategy, for instance by the renowned Support Vector Machines (SVM) which optimise both accuracy and complexity through the use of structural risk minimisation framework as a part of its training process (Cortes and Vapnik (1995),Vapnik (2013)).

However, model complexity is just one criterion among many other criteria that can affect the choice of predictive systems. In order to improve the performance of the prediction, ensemble learners are often used (Polikar (2006)). While these systems are substantially more sophisticated and have higher computational costs than a single predictor, combining different predictors can increase the accuracy of the overall system, provided enough diversity among the components of such systems i.e. their capacity to complement each other is sufficiently observed. (e.g., Jacobs (1995), Meir (1995), Opitz and Shavlik (1996a) and Tumer and Ghosh (1996)). Though encouraging diversity among the base predictors of ensembles is widely acknowledged as an important issue in improving the ensemble performance (Bi (2012)), there has not been found a fundamental connection between the current diversity measures and the improvement of the prediction accuracy in general (Brown and Kuncheva (2010), Bi (2012) and Kuncheva and Whitaker (2003)).

In addition to improving the accuracy of the prediction, combining multiple classifiers has been linked to the ability of the system to perform well with noisy data (Ho et al. (1994)). Associating robustness to noise with Multi-Components, Multi-Layers Predictive System (MCMLPS) has been also traditionally linked to the diversity among the system base models: combining diverse models can improve the generalization ability of the system due to their complementary behaviour and allow the system to be less subjected to overfitting noisy data (Teng (1999) and Sáez et al. (2013)).

Unfortunately, defining effective measurements for the above- mentioned criteria is not always feasible in all machine learning methods. Thus, when measuring these criteria is infeasible, alternative mechanisms must be put in place to ensure that they are considered during the different stages of the predictive system design cycle, such as: a) using regularisation terms to penalize models with high complexity (Barron (1991)) and b) training

the ensemble base models on slightly different subsets of the data to encourage the diversity among them (Breiman (1996) and Rodriguez et al. (2006)).
Nevertheless, in cases when suitable measures for the multiple optimization criteria can be defined, Multi Objective Optimisation (MOO) approaches can be effectively used to maintain the desired trade-off among these criteria (Marler and Arora (2004) and Jin (2006)). There are a number of optimisation approaches for combining these criteria. One common approach is to use scalarization function which combines them into a single weighted sum and find the best possible model, such as: in neural networks regularisation (Braga et al. (2006)) and in creating interpretable fuzzy rules (Jin (2000)). Another approach is to use Pareto-based MOO techniques to find a set of non-dominated solutions (models) that trade-off a set of conflicting criteria; such as: trading-off the accuracy versus the complexity of a radial basis function (Hatanaka et al. (2003)), trading-off false negative/false positive rates versus the number of support vectors to reduce the computational complexity of an SVM (Suttorp and Igel (2006)) or developing a cost sensitive decision tree (Zhao (2007)) to cite a few.
In summary, considering multiple criteria in optimising complex predictive system entails unresolved challenges connected with how to find suitable measures of such criteria and how to include them in the design cycle of the predictive system. Also as was mentioned above, previous approaches typically focus on pairs of criteria to optimise in which one of them is the accuracy of the prediction (such as balancing accuracy and model complexity in (Yu et al. (2006)), or studying the effect of diversity with respect to the accuracy of the system (Cunningham and Carney (2000)). However, to the best of our knowledge studying the interaction among multiple criteria and the effect they have on each other have not previously been fully investigated; and will be the main focus of this thesis.

1.2 Project Description and Goals

This thesis emphasises the importance of including multiple criteria in the design process of predictive systems and studies the interaction among them. Furthermore, it compares the different optimisation approaches used to trade-off these criteria. The main goals of this work are:

- To identify the criteria used for evaluating the performance of a predictive system from multiple angles and to define effective measures for them when feasible.

- To develop strategies to include the selected criteria either directly in the optimisation process or indirectly in the design process of the predictive system.
- To study the interactions among the included criteria and identify the main factors/decision points in the MCMLPS that affect them.

This work examines the relations among the accuracy of the prediction, model complexity, algorithmic complexity, diversity and robustness. Furthermore, this project compares the efficiency of different optimisation techniques which can be used when multiple criteria are considered (Chapter 2 and 3). It examines the advantages and drawbacks in the cases when the criteria are combined in a single scalarized equation or when a trade-off among them is maintained (Chapter 3).

The initial experimental work considers the inclusion of the first three criteria (namely, the accuracy, model complexity and algorithmic complexity) in the optimisation process of the MCMLPS base components (Chapter 3). These experiments focus on different optimisation techniques used to include multiple criteria. Both Pareto-based MOO and scalarized MOO are used to generate the predictive models, and the performance of the resultant models is assessed and compared. The aims of these experiments are to examine the advantages and drawback of including multiple criteria and to compare the efficiency of the generated models using the two optimisation approaches.

Taking the above into consideration, a novel locally trained MCMLPS is introduced in Chapter 4. In this system, the diversity among the base models is maximized by training them on local disjoint sets of data and/or subsets of features. The data is divided into local regions using two approaches: an unsupervised approach which uses the feature similarity depending on their pairwise squared correlation and a supervised approach which is based on mutual information theory. This work is further developed to investigate the relation between the models diversity and their robustness to noise by introducing six fusion methods used to deliver the final prediction for the proposed MCMLPS (Chapter 5).

Finally, the interactions among the accuracy, diversity and robustness of this system is examined and the overall performance of the system is compared to a number of ensemble methods (Chapter 5). Our results indicated that the locality and high diversity among the components of the proposed system can provide a robust framework for designing complex systems in noisy environments.

1.3 Contributions

The following points summarise the main contributions in this work:

- A comprehensive theoretical study of predictive systems in terms of their architectures, evaluation criteria and optimisation approaches. This study focuses on identifying measures for the principle criteria that affect the performance of machine learning systems and whether universal measures can be defined for these criteria (Chapter 2).
- Conducting a new experiment in a representative, specifically designed case study which compares two MOO approaches and highlights the cost and benefits obtained from including multiple criteria in the optimisation of machine learning models in different classification settings (Chapter 3).
- A novel locally trained MCMLPS is next proposed (Chapter 4), where the locality of the system is introduced using two approaches:
  - An unsupervised approach is used to split the data into disjoint subsets that are assigned to a set of local regions. The locality is determined using the pairwise squared correlation of the features. Then the base predictors of the MCMLPS are trained on these local regions. A particular benefit of MCMLPS is that, since it trained the local regions on disjoint subsets of the data, the diversity among its components is maximised.
  - A supervised approach is used to split the data into local regions using the conditional mutual information of their features. In this approach the base models are trained on subsets of the features for all available data.
- An analysis of the robustness of the new MCMLPS in comparison with well-known ensemble methods; and its relation to the diversity of the proposed system in noisy environments (Chapter 5).
- The identification of the main *decision points* (in the design and weighing of the MCMLPS) which influence the robustness, diversity and accuracy of the prediction (Chapter 5). These decision points are found to be:
  - Data partitioning and model training.
  - Weighing the prediction of the base models/ensembles.
  - Selection/fusion of the base models/ensembles.

1.4 Thesis Organization

Chapter 2 provides an overview of the learning process in predictive system. In Section 2.2 different architectures of predictive systems are explained, from simple single predictor to complex multi-layer systems. Furthermore a survey of the predictive system evaluation criteria and their possible measurements is given in Section 2.3. Section 2.4 provides an overview of the optimisation approaches used to optimise single as well as multiple criteria of the predictive system.

In Chapter 3 the general methodology is discussed, starting with the design cycle of MCMLPS in Section 3.2. Next, Section 3.3 introduces a comparative case study specifically designed to take into account the optimisation of predictive models using prediction accuracy, model complexity and algorithmic complexity. This new study compares the base models of MCMLPS that are generated from optimising the above mentioned criteria using scalarized multi-objective optimisation and Pareto-front multi-objective optimisation. Furthermore, this section highlights the limitations associated with each of the two optimisation approaches.

Chapter 4 introduces a novel locally trained MCMLPS, where its base models are trained on disjoint subsets of the data and/or subsets of the features. The general architecture of the proposed system is given in Section 4.2. Section 4.3 compares the design cycle of MCMLPS with that of the rotation forest algorithm. Next, the methodology followed in designing the proposed system is given in Section 4.4 along with a detailed description of the metrics used to define the locality of the data, the experimental settings and the results.

Chapter 5 expands the work presented in Chapter 4 by testing the MCMLPS in noisy environments. The different types of noise and their effect on the performance of the system are explained in Section 5.2, while balancing the robustness and flexibility of machine learning models are the focus of Section 5.3. The first part of the experimental work in this Chapter is introduced in Section 5.4 where the proposed MCMLPS is tested in noisy environments and its performance is compared to other well-known ensemble methods. The second part of the experimental work, presented in Section 5.5, introduces six fusion methods to combine the base predictors of the MCMLPS and studies the effect of changing the fusion methods on the performance of the proposed system.

Finally, Chapter 6 concludes the thesis, summarising the main findings and contributions of this work and indicating directions for future research.

1.5 Publications resulted from this work

- Al-Jubouri, Bassma, and Bogdan Gabrys. ”Multicriteria approaches for predictive model generation: a comparative experimental study.” Computational Intelligence in Multi-Criteria Decision-Making (MCDM), 2014 IEEE Symposium on. IEEE, 2014.
- Al-Jubouri, Bassma, and Bogdan Gabrys. ”Local Learning for Multi-layer, Multi-component Predictive System.” Procedia Computer Science 96 (2016): 723-732.
- Al-Jubouri, Bassma, and Gabrys, Bogdan. Diversity and Locality in Multi-Component, Multi-Layer Predictive Systems: A Mutual Information Based Approach In Advanced Data Mining and Applications: ADMA 2017, Singapore, November, 5-6, 2017. Springer International Publishing.
- Al-Jubouri, Bassma, and Gabrys, Bogdan. Interaction between robustness and diversity in multi-layer, multi-component predictive systems Information Fusion, 2018. (Submitted).

Chapter 2

Predictive Systems: Representation, Evaluation and Optimisation

2.1 Introduction

The ability of machines to think was first questioned by Alan Turing in (Turing (1950)). In this paper the Turing test was introduced, where in a simple test, a judge (human) is asked to distinguish between a machine and a real person depending on their answers to particular questions. By 1959 machine learning was defined as the ability of a computer to perform a task that it has not been explicitly programmed to do, in a similar way to the learning behaviour in humans or animals (Samuel (1959)). In recent years, formal descriptions of machine learning can be found in (Duda et al. (2012), Theodoridis et al. (2010), Michalski et al. (2013) and Anzai (2012)). For example in (Duda et al. (2012) machine learning is defined as the estimation of the parameter values for a model using sample data in order to optimise a criterion function.
According to (Budka (2010)) a predictive system S can be trained to approximate an existing unknown function $M : \mathbb{R}^d \rightarrow \mathbb{R}^c$ which maps a d-dimensional input space $\mathcal{X}$ into a c-dimensional output space $\mathcal{Y}$:

$$M{:}\mathcal{X} \rightarrow \mathcal{Y} \tag{2.1}$$

In order for a predictive system S to provide an approximation of the mapping M a learning algorithm is used to tune the system parameters. This algorithm learns from

examples, where the training data D that consist of N instances is used to provide the sufficient information for the learning algorithm:

$$D = \{(X, \hat{Y})\} = \{(x_1, \hat{y}_1), (x_2, \hat{y}_2), ..., (x_N, \hat{y}_N)\} \tag{2.2}$$

Where $x_i \in \mathbb{R}^d$. Due to the limitation in the precision of the data collection process, the predictive system learns to map the input to a predicted output $\hat{Y}$ (where $\hat{Y} \in \mathbb{R}^c$) rather than the actual output Y, where $\hat{y}_i \neq y_i$ instead:

$$\hat{y}_i = y_i + \epsilon \tag{2.3}$$

Where ϵ is a zero mean random noise with expectation of $E[\epsilon_i] = 0$ (Duda et al. (2012) and Budka (2010)). The type of the output $\hat{Y}$ can be continuous (regression problem) or discrete (classification problem). The new mapping of the predictive system is given below:

$$S{:}\mathcal{X} \rightarrow \hat{\mathcal{Y}} \tag{2.4}$$

In order to measure the accuracy of this system, an error function is used. A general formula error function can be given in:

$$error = \frac{1}{N} \sum_{i=1}^{N} f(y_i, \hat{y}_i) \tag{2.5}$$

Generally $f(y, \hat{y}) = f(y - \hat{y})$ in regression problems, while in classification problems $f(y, \hat{y}) = f(1 - \delta_{(y-\hat{y})})$ where $\delta_{(y-\hat{y})}$ is the Kronecker delta function given as:

$$\delta_{(y-\hat{y})} = \begin{cases} 1, & if y \neq \hat{y} \\ 0, & if y = \hat{y} \end{cases} \tag{2.6}$$

Depending on the availability of the data and the output, mainly there are four forms of learning that can be identified Duda et al. (2012):

- **Supervised learning**: this type of learning is used to infer a function learned from labelled training data. The training data consists of pairs of input-output data. The inferred function can be used to map new input data to its correct output value. This type of learning is also known as learning with a teacher. The same behaviour observed in humans and animals is called concept learning. The mathematical

representation of the predictive systems discussed above is mainly provided for supervised learning.

- **Unsupervised learning**: in this type of learning only the input data is provided to the learning algorithm without its output (unlabelled data). The learning process is associated with data density estimation or clustering, where the predictive system forms clusters or natural grouping from the input data. Similar data are assigned to the same cluster or group and it is assumed that they share the same label. Different clustering methods lead to different sets of clusters and it is often that the number of clusters is predefined by the user. The accuracy of this learning approach depends to a large extent on the choice of the metric used to measure the similarity. This type of learning is also known as learning without a teacher.
- **Semi-supervised learning**: this type of learning falls between supervised and unsupervised learning, where the labels are provided for only part of the input data. The acquisition of unlabelled data is often inexpensive. Though it can be ignored in the learning process (making the problem a supervised learning problem), yet using this data can improve the prediction of the system. Examples on how the unlabelled data can be included in the learning process are: assuming that the points that are close to each other share the same label (smoothness assumption), or that the data tends to form discrete clusters and that the points in the same clusters share the same label (cluster assumption).
- **Reinforcement learning**: in this type of learning the predictive system performs a series of actions in order to maximize some notion of accumulative reward. Rather than having an input-output pairs of data, the only information available to the system is whether the final prediction is right or wrong (a binary feedback). Thus in binary classification problems with equal cost of error reinforcement learning is equivalent to supervised binary classification. The feedback can be provided after a few steps or in extreme cases after a long series of actions. This type of learning is also known as learning with a critic, where the critic says only if the prediction is correct or not without specifying how it is incorrect.

Regardless of the learning type, in general, the learning process in predictive systems can be decompose into three components (Domingos (2012)):

$$Learning = Representation + Evaluation + Optimisation \tag{2.7}$$

The first component considers model representation which involves choosing the type and architecture of the predictive system (model selection) as well as the representation of the data (data collection, preparation and partitioning). The second component defines the criteria used to evaluate the predictive system performance, which can be a single or multiple criteria. Once these criteria are identified and measured, they are used to optimise the performance of the predictive system. The three components of the learning process along with their main operations are shown in Figure 2.1.

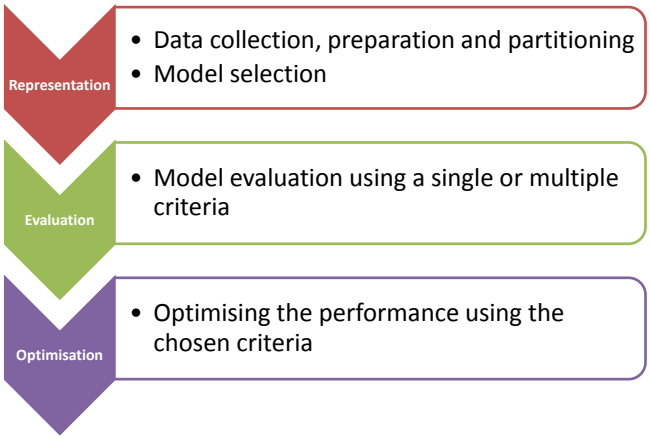


Figure 2.1: *The decomposition of the learning process (based on Domingos (2012))*

This chapter discusses the predictive systems, their evaluation criteria and optimization approaches. It starts, in Section 2.2, by describing the architectures of predictive systems from simple single model to sophisticated pool of competing predictors. Section 2.3 looks at the evaluation criteria and their possible measures. Finally, the optimisation methods used for balancing and trading-off these criteria are discussed in Section 2.4.

2.2 Predictive Systems Architectures

Predictive systems can have various types of architectures. These architectures can vary from single model to complex multiple competing structures. The following sections

provide explanations and illustrations of the different architectures of predictive systems.

- **Single Predictor**:

  This architecture consists of a single predictor with a possible one or more pre-processing and post-processing unit(s). The complexity of this model can vary from a simple linear regression method to complex methods like Support Vector Machines (SVM) (Cortes and Vapnik (1995)). Figure 2.2 shows a simple illustration for this architecture, where $\hat{f}(x)$ is a function that the predictive model is learning to approximate the actual function $f(x)$ which generates the data D, PPM is the pre-processing method.

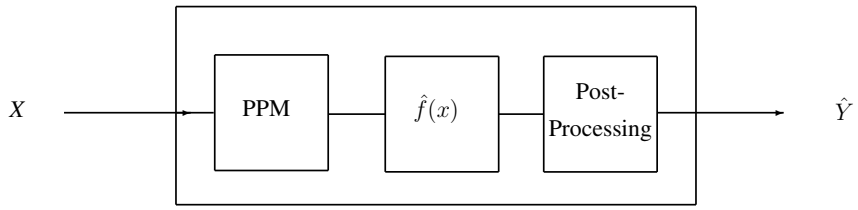


Figure 2.2: *A Simple illustration for a single predictor consisting of a pre-processing unit, a prediction function and a post-processing unit*

A predictive model can be trained to have different levels of complexities, for example, having different number of hidden layers and/or hidden units in an artificial neural network. A main challenge is to choose the appropriate model architecture, such that, the model is not too simple or too complicated to represent the data. One approach for identifying the required level of complexity to match the prediction problem is to decompose the generalisation error into bias and variance components. Trading off these two components properly can reduce the overall generalisation error and prevent overfitting the data. Basically, the error due to the bias term represents the difference between the prediction of the model and the actual output. On the other hand, the error due to the variance represents the variability in the model prediction for a given data point. In order to obtain a good low generalisation error both bias and variance should be minimised. However, as the two components are conflicting a trade-off between them should be maintained. The bias-variance dilemma is illustrated in Figure 2.3.

Initially the bias-variance decomposition was developed for least squares regres-

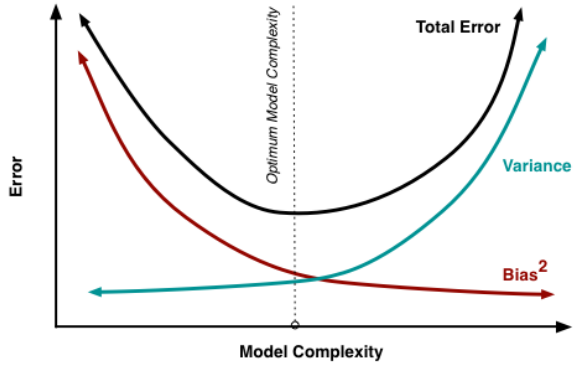


Figure 2.3: *The bias-variance dilemma. Adopted from (Fortmann-Roe (2012))*

sion, however, in recent years other forms of decomposition for binary classification and probabilistic classification has been introduced (Domingos (2000)). The original mathematical form of the bias-variance decomposition for the mean square error is given below.

In the following equations let us assume that $(\hat{Y}) = \hat{f}(x)$. The expected Mean Square Error (MSE) between the model prediction $\hat{Y}$ and the actual output Y over an infinite number of data sets of size N is given as Bishop (1995):

$$E_D[MSE(\hat{Y}, Y)] = \frac{1}{N}\sum_{i=1}^{N} E_D[(\hat{y}_i, y_i)^2] \tag{2.8}$$

This MSE formula can be decomposed into the bias and variance components and noise (Bishop (1995)).

$$E_D[MSE(\hat{y}_i, y_i)] = \underbrace{[(y_i - E_D[\hat{y}_i])^2]}_{bias} + \underbrace{E_D[(\hat{y}_i - E_D[\hat{y}_i])]^2}_{varinace} + \underbrace{E_D[\epsilon^2]}_{noise} \tag{2.9}$$

The aim of introducing bias-variance decomposition in this section is to show that even for a single model type different architectures can result in different levels of performances. This decomposition is viewed from the selection of model architecture point of view rather than from the evaluation of the performance point of view. There are more practical approaches that can be used to measure and control the model complexity. These approaches will be explained in Subsection 3.2.2.

- **Ensemble of predictors:**

  An ensemble is a predictive system consisting of a number of base predictors combined together using a combination method to provide the final prediction. The combination method is sometimes known as a fusion method. In the past decades ensemble learning has been an active area of research in machine learning. Many well-performing ensemble learning algorithms have been introduced such as: Boosting (Freund and Schapire (1996)), Bagging (Breiman (1996)), and stacked generalization (Wolpert (1992)). In literature, it has been shown that using an ensemble of predictors can often improve the generalization performance compared to that of a single predictor. The conditions for this improvement are for the base predictors to be diverse (their error correlation is reduced) and that they have a reasonable performance level (Jacobs (1995), Meir (1995), Opitz and Shavlik (1996a) and Tumer and Ghosh (1996))

  An ensemble with diverse models can have better performance due to the complementary behaviour of its components (Xue et al. (2006)). The performance of an ensemble that consists of identical predictor will not be better than any of its base components. Ensemble learning can be viewed as combining multiple predictive models in order to explore the space and benefit from the models complementary predictive characteristic, thus the models must be diverse so that their fusion can have better performance than their individual performance.

  The mathematical justification for encouraging diversity was first explored in the ambiguity decomposition (Krogh and Vedelsby (1995)), where the ensemble diversity is linked to the mean square error. In this study, Krogh and Vedelsby proved that:

  "at a single data point the quadratic error of the ensemble estimator is generated to be less than or equal to the quadratic error of the component estimator"

  Equation 2.10 shows the ambiguity decomposition for the squared error:

$$(\hat{f}_{ens}(x) - f(x))^2 = \sum_i w_i(\hat{f}_i(x) - f(x))^2 - \sum_i w_i(\hat{f}_i(x) - \hat{f}_{ens}(x))^2 \quad (2.10)$$

  where $\hat{f}_{ens}(x)$ is the fusion function, $f(x)$ is the target, $\hat{f}_i(x)$ is the function for the base predictors and w_i's are the weights and they sum up to one. This equation shows that the squared error of an ensemble $\hat{f}_{ens}(x)$ is equal to the

weighted averaged squared error of the individual base predictors minus a term which quantifies the diversity of the ensemble. This term measures the correlation between the base predictors output and the overall ensemble output.

The main limitation of the ambiguity decomposition is that it can be applied only to linearly weighted regression problems and it needs to define in advance a set of optimal weights as well as a set of models that are both accurate and diverse.

Depending on the type of the base predictors, whether they are similar or different, ensemble learning can be classified into: homogenous ensembles and heterogeneous ensembles (hybrid ensembles) (Whalen and Pandey (2013) and Woźniak et al. (2014)). In homogenous ensembles the base predictors are all of the same type, for example all the base predictors are Decision Trees (DTs) or Neural Networks (NN). An illustration of this type of ensemble architecture is shown in Figure 2.4. On the other hand, in heterogeneous ensembles, models of

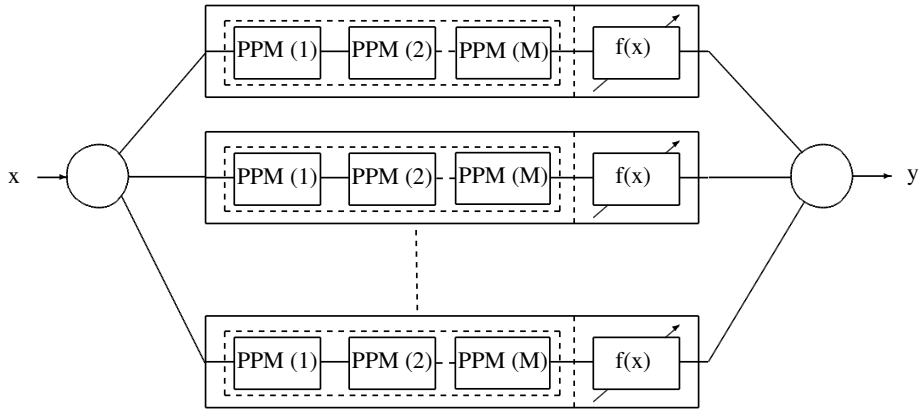


Figure 2.4: *An illustration for a homogeneous ensemble consisting of multiple predictors of the same type*

various types are combined to provide the final prediction of the system. There are a number of challenges associated with this type of predictive systems. These challenges include (Woźniak et al. (2014)): a) choosing the appropriate models to be combined, b) tuning the models parameters and understanding the effect these parameters have on the individual models performance as well as on the overall performance of the system, and c) choosing the appropriate combining method for the base predictors. The final prediction can be obtained using either selection or fusion methods (Woods et al. (1997) and Kuncheva (2004)). In classifier selection, the base predictors are trained on local regions of the feature space. The locality

of the data can be determined using a number of metrics such as distance metric, correlation or mutual information among others. Depending on the metric used, the final prediction of such ensemble can be obtained from one or more predictors (Bloch (1996) and Roli et al. (2001)). On the other hand, in classifier fusion, the outputs of all the base predictor ares considered in the final prediction, examples of this approach are bagging and boosting methods. Figure 2.5 shows a simple example of heterogeneous ensemble.

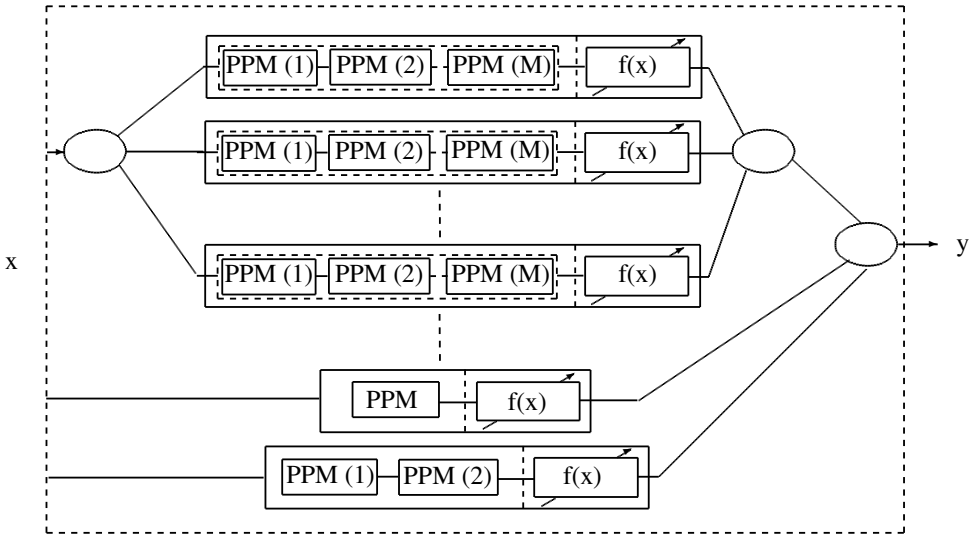


Figure 2.5: *An illustration for a heterogeneous ensemble consisting of multiple predictors of different types*

A more complex architecture of heterogeneous ensembles is to have a pool of competing, possibly complex predictors. In this case, multiple predictive systems of different complexities and types are organised in a pool of competing solutions. The predictors can have varying complexities from single predictors to MLMCPS systems. An illustration of this architecture is shown in Figure 2.6.

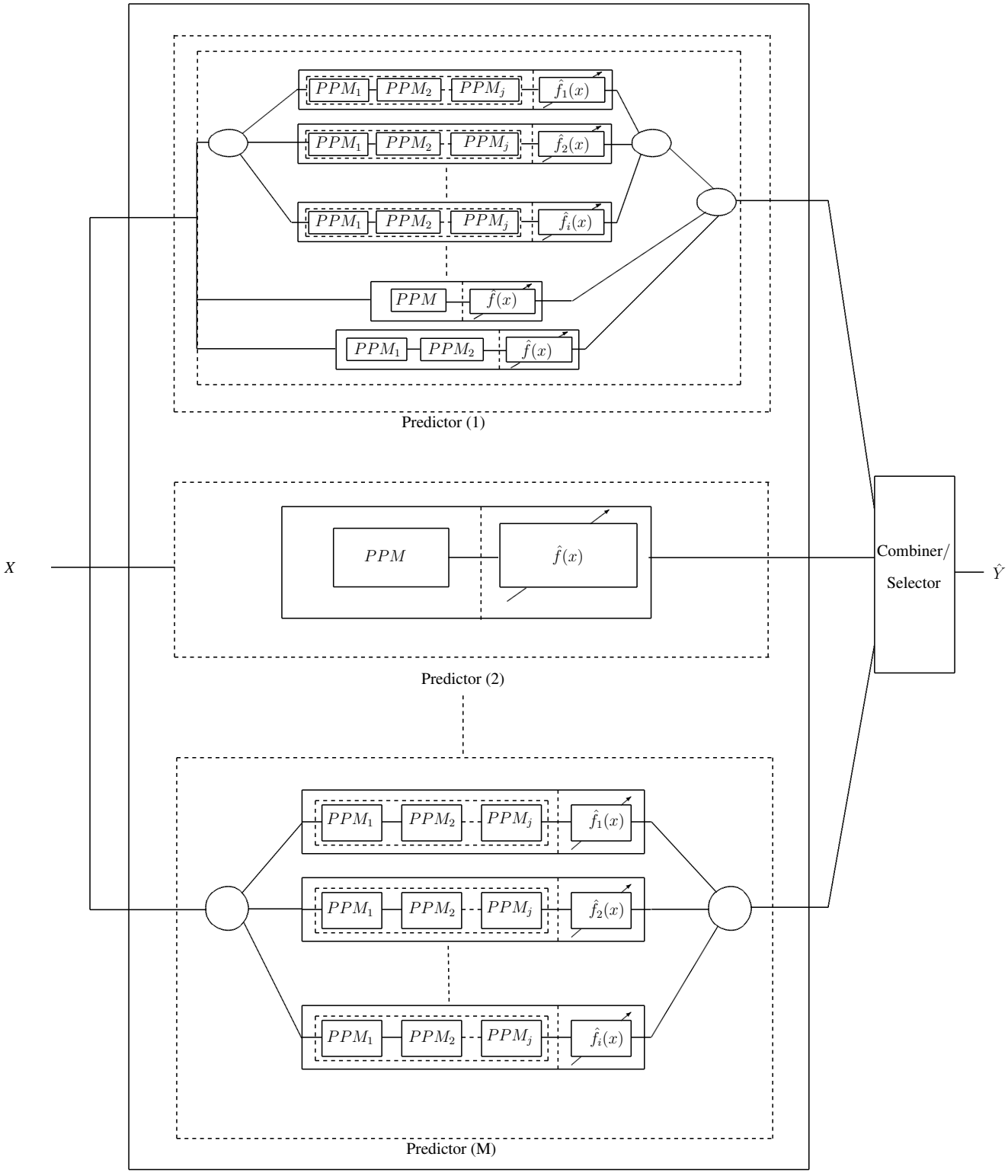


Figure 2.6: *An illustration for a pool of competing predictors/ensembles*

2.3 Predictive Systems Evaluation

The performance of both single model and ensemble architecture can be evaluated based on a number of criteria. These criteria include: accuracy, complexity, robustness, adaptation and transparency. This section considers these criteria and their most common well-known measures.

2.3.1 Accuracy

The accuracy of prediction can be calculated, depending on the prediction problem using one of the following measures.

- **Numerical measures:**
  The following points summarise the main measures for calculating the error E (Hyndman and Koehler (2006)):
  1. Mean Square Error (MSE):
  $$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{2.11}$$
  2. Root Mean Square Error (RMSE):
  $$RMSE = \sqrt[2]{MSE} \tag{2.12}$$
  3. Sum of Square Regression (SSR):
  $$SSR = \sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{2.13}$$
  4. Mean Absolute Error (MAE):
  $$MAE = \frac{1}{N}\sum_{i=1}^{N} \mid (\hat{y}_i - y_i) \mid \tag{2.14}$$

5. Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100\%}{N} \sum_{i=1}^{N} \mid \frac{(\hat{y}_i - y_i)}{y_i} \mid \tag{2.15}$$

6. Root Mean Square Percentage Error (RMSPA):

$$RMSPE = \sqrt[2]{[\frac{100\%}{N} \sum_{i=1}^{N} \frac{(\hat{y}_i - y_i)}{y_i}]^2} \tag{2.16}$$

Measures 1-4 are scale-dependent measures, since they depend on the scale of the data. They are mainly used to compare different algorithms applied to the same data set or to data sets of similar scales. Nevertheless, these measures should not be used when the predictive systems are compared across data-sets of different scales. On the other hand, measures 5 and 6 are scale-independent and often used to compare algorithms across different data-sets. However, both measures 5 and 6 are undefined or infinite when the actual output is zero, since the error will be divided by zero in this case.

- **Confusion matrix:**
  Confusion matrix (or contingency table) is a table that illustrates the classification performance of the predictive system. Table 2.1 shows the confusion matrix of a binary classification problem (Alpaydin (2014)).

Table 2.1: *Confusion matrix for binary classification problems*

	actual positive	actual negative
predicted positive	True Positive (TP)	False Negative (FN)
predicted negative	False positive (FP)	True Negative (TN)

Assuming that there are two classes: positive class and negative class, the elements shown in the above matrix are (Alpaydin (2014) and Fawcett (2006)):

TP (True Positives) is the number of examples correctly classified as positives.

TN (True Negatives) is the number of examples correctly classified as negatives.

FP (False positives) is the number of negative examples incorrectly classified as positives.

FN (False negatives) is the number of positive examples incorrectly classified as negatives.

For this problem the classification error can be calculated using the following equation:

$$E = \frac{FP + FN}{FP + FN + TP + TN} \tag{2.17}$$

Several metrics can be derived from the confusion matrix, such as:

Precision Also known as the positive predictive value (PPV). It measures the number of the correctly classified positives divided by the total number of positive examples.

$$Precision = \frac{TP}{TP + FP} \tag{2.18}$$

Recall Also known as the sensitivity, it measures the proportion of the positive examples that are correctly identified.

$$Recall = \frac{TP}{TP + FN} \tag{2.19}$$

Specificity It measures how well the classifier detects the negative examples.

$$Specificity = \frac{TN}{TN + FP} \tag{2.20}$$

F-measure (F-score) is an accuracy test that can be calculated using a weighted average of the precision and recall as shown in the following equation:

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.21}$$

The best value a classifier can achieve in this measure is 1 and the worst value is 0.

When the number of classes (k) exceeds 2; the confusion matrix becomes a $(k \times k)$ matrix (Alpaydin (2014)). The main diagonal of the matrix contains the correctly classified examples and the off diagonal elements contain the examples that are

misclassified. Ideally all off-diagonal elements should be 0.

- **ROC and AUC:**
  Receiver Operating Characteristic (ROC) is a 2-dimensional graph used to visualise the classifier performance in binary classification problems (Kubat et al. (1998)). Figure 2.7 shows an example of the ROC curve, where the y-axis of this graph represent the TP_{rate} (recall) and the x-axis of the graph represent the FP_{rate} $(1 - specificity)$. This curve maintains a trade-off between the benefits (true positives) and the cost (false positives). An Ideal classifier will have a $TP_{rate} = 1$ and $FP_{rate} = 0$. The closer the classifier is to the upper-left corner the better is its accuracy.

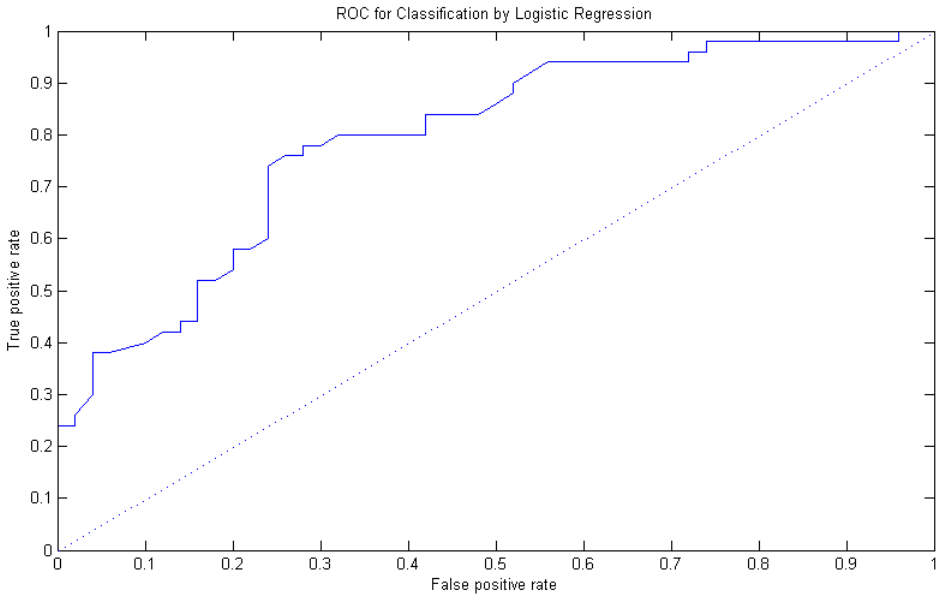


Figure 2.7: *The ROC curve for classifying the Virginica class in the Fisher iris data set using logistic regression*

The worst case in binary classification is when the classifier performance lies on the main diagonal. In such a case it will have an accuracy value of 50%. However, the performance of classifiers that go below the main diagonal can be improved by flipping their decision (Alpaydin (2014)). In order to obtain a single value that represents the accuracy of the classifier, the Area Under the Curve (AUC) is calculated. An ideal classifier will have an AUC=1.

- **Weighted accuracy (cost sensitive) measures:**
  As explained before, the confusion matrix distinguishes different types of error. In some applications different costs are associated with misclassification errors. For instance, the cost of misclassifying a simple injury as deadly is much lower than the cost of misclassifying a deadly injury as a simple one (King et al. (1995)).
  For such applications a cost matrix is constructed. This matrix provides the cost of each type of error. For example, the cost matrix of a binary classification problem is given as:

		actual positive	actual negative
	predicted positive	$Cost(0,0)$	$Cost(0,1)$
	predicted negative	$Cost(1,0)$	$Cost(1,1)$

  The optimal prediction in this case can be calculated using the following equation (Elkan (2001)):

$$L(c,i) = \sum_j P(j \mid c) Cost(i,j) \tag{2.22}$$

  Where L is a function which calculates the loss, i is the index for the predicted class, j is the index of the true class, $P(j|c)$ is the probability of the class j being the true class c, and $Cost(i,j)$ is the cost of the prediction.

2.3.2 Complexity

Complexity can be divided into two categories: model complexity and algorithmic complexity (Russell and Norvig (2010)). Model complexity is the complexity of the final trained model, which can be obtained using different training algorithms. On the other hand, the algorithmic complexity is the complexity of the algorithm used to train the model. The following discussion highlights the differences between these two concepts. Consider Table 2.2, the second column shows various types of predictive models that can be trained to fit a linearly separable classification problem. Though the generated models are all linear models, however, their complexities and the complexities of their training algorithms are widely varied. Increasing the complexity of the classification problem (as shown in the third column) will require changes in the complexities of the models as well as their training algorithms. The algorithmic complexity of the training model plays an

Table 2.2: *Learning methods used to generate linear and quadratic predictive Models*

Model type	Linear Model	More Complex Model
Training algorithm	Linear regression	Second or higher order polynomial
	SVM with linear kernel	SVM with nonlinear kernel
	Linear perceptron	MLP
	DT with one node	DT with more than one node/level
	Rule base system with single rule	Rule base system with multiple rules

important role in dynamic environments, as the model may need to be retrained repeatedly.

Meanwhile, the complexity of the developed model should be chosen such that a complex model is not chosen for a problem that can be solved using a simpler model (Blumer et al. (1987)).

Figure 2.8 shows an example where three functions of different complexities are used to model a given data set. The first function is a simple linear fit, and it represents the case where the model has lower complexity than what the application requires. The middle function shows a model with moderate complexity and it shows a good representation of the data. Finally the third function shows a model that has higher complexity level than what the application requires.

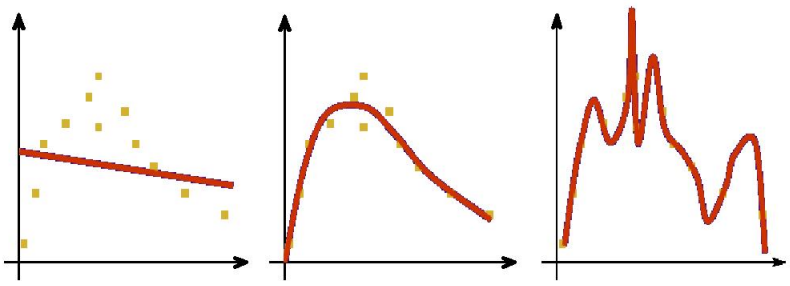


Figure 2.8: *A data set fitted with three functions of increasing complexity. Adopted from Gunn (2012).*

2.3.2.1 Algorithmic Complexity Measures

Algorithmic complexity can be measured using the following methods:

- **Kolmogorov complexity**

  Though Kolmogorov complexity is a theoretically well-defined method for measuring the algorithmic complexity, it is not applicable in practise. This method defines the complexity of an algorithm as the length of the shortest Turing machine program that can represent this algorithm. The Kolmogorov complexity is given below (Jankowski and Grabczewski (2011)):

$$\Omega_k(Pr) = min_{pr}(l(Pr) : \text{ program pr prints Pr}) \tag{2.23}$$

  Where $\Omega_k(Pr)$ represents the Kolmogorov complexity and $l(Pr)$ represent the length of the program. Nevertheless, the search space for this problem is unlimited and the program execution time is also unlimited, which makes finding the program with the shortest length an unsolvable problem. This is known as the halting problem and can be stated as (Reed (2005)):

  > *"Given a description of an arbitrary computer program, decide whether the program finishes running or continues to run forever"*

- **Levin Universal Search (LUS)**

  LUS introduces a time-bound on the Kolmogorov complexity and by that it introduces a computable version of the Kolmogorov complexity, the LUS equation is (Jankowski and Grabczewski (2011)):

$$\Omega_l(Pr) = min_{pr}(l(Pr) : \text{ program pr prints pr in } t^{pr}) \tag{2.24}$$

$$\Omega_l(Pr) = l(Pr) + log(t^{pr}) \tag{2.25}$$

  Where t^{pr} is the time required to finish the program.

  Nevertheless, solving the LUS is an NP-hard (Non-deterministic Polynomial-time hard) problem, where, an NP problem is a problem for which a solution can be verified in polynomial time using a deterministic Turing machine. Meanwhile, an NP-hard problem is a class of problems which are at least as hard as the hardest problem in NP. Due to this, in practise it is impossible to find the exact solution for this optimisation problem (Jankowski and Grabczewski (2011)).

- **Asymptotic analysis and big O notation**

  Asymptotic analysis can be used to measure the algorithmic complexity of a

learning method. It measures the number of operations performed by an algorithm and the size of its input (Russell and Norvig (2010)). However, finding the exact number of operations carried out by an algorithm is often non-trivial. In such situation the number of operations required to solve the worst case or the average case in the problem is considered. This approximation is noted as the big O function. The most common types of O functions are:

- Constant
- Logarithmic $(log\ n)$
- Linear(n)
- N-log-N $(n\ log\ n)$
- Quadratic (n^2)
- Cubic (n^3)
- Exponential (2^n)

The main limitation of asymptotic analysis is that, though it provide a mechanism for measuring the time and memory usage of algorithms, it does not consider the type of the problem, the programming language or the machine limitation (Russell and Norvig (2010)). In practise, the algorithmic complexity is often evaluated by measuring the execution time and memory usage and it can be presented either as a constraint or an additional criterion in the optimisation process.
The execution time represents the training time, testing time, and the time required to deliver a single prediction. It can be measured in (msec., sec., min.,etc.) depending on the problem. Meanwhile, the memory usage represents the total amount of memory required to store the structure of the algorithm and the memory required to train the model. It can be measured in (bytes, Mbytes,etc.).

2.3.2.2 Model Complexity Measures

In order to compare models complexity across models from different fields a general definition for the model complexity is required (like the big O notation used for the algorithmic complexity).

- **Number of free parameters**

  One way for measuring the model complexity is to determine the number of free parameters required by the model. This value can be found for all machine learning algorithms and can be compared across models of different types. However, the difficulty of finding these parameters varies for different machine learning models. For example, the difficulty of finding the support vectors in SVM is not the same as that of finding the weights of an ensemble.

  Furthermore, this approach does not consider the internal complexity of the model. For example, if two Neural Networks (NN) of the same structure (the same number of hidden layers and hidden units) were considered, according to this approach they will be treated as if they have the same number of free parameters (same number of weights). However, they can have activation functions of varying complexities. For instance, one could have a simple step function, while the other could have a sigmoid function. Such internal complexities are not considered in this approach.

- **Bounding the generalisation error**

  Another approach to consider the model complexity is to introduce bounds on the generalisation error. Models of different types can be compared based on their generalisation error, and the model complexity is involved in the comparison in an indirect way. One example of controlling the model complexity through the generalisation performance, is to stop training the model when the error generated on the validation set starts to increase, i.e. when the model starts to overfit the training data and the complexity is higher than what is required.

  The above methods can be used to compare complexities (directly or indirectly) across different types of predictors. However, there are other methods for controlling the model complexity with respect to the accuracy of models of the same type. Examples of these methods are:

  - **Structural Risk Minimization (SRM)**

    Structural Risk Minimization (SRM) principle was developed in 1974 by Vapnik and Chervonenkis. The SRM principle defines a function that restricts the model complexity with respect to its empirical error. The complexity is

measured using Vapnik Chervonenkis dimension (VC-dimension); it can also be measured using a generalised version of the VC-dimension known as the fat shattering dimension (Kearns and Schapire (1994)).

The SRM principle shows that the upper bound of the machine expected risk is restricted by two factors. These factors are the empirical risk (the error) and the VC-dimension (the complexity). The SRM follows the worst case scenario by optimizing the upper bound of the expected error rather than the expected error itself.

– **Bias-Variance-Complexity decomposition**

Another approach for comparing the models based on their accuracy and complexity is the bias-variance-complexity decomposition (introduced in Yu et al. (2006)). This method attempts to define a selection criterion which takes into account the bias, the variance and the model complexity and aims to minimize this criterion. The minimization process is considered as a Multi-Objective Optimisation (MOO) problem, it trades off the bias and variance of the models for a given (or an appropriate) model complexity. The model complexity is measured using the number of parameters or degrees of freedom. The selection criterion is (Yu et al. (2006)):

$$selection\ criteria = [(n+\Omega)/(n-\Omega)].[(\bar{\mathrm{h}}(x)-f(x))^2+E[(\hat{f}(x)-\bar{\mathrm{h}}(x))^2]] \tag{2.26}$$

Where n is the size of the training data, Ω is the number of free parameters, $\bar{\mathrm{h}}$(x) is the closest value in the hypothesis space to the target, $f(x)$ is the target function and $\hat{f}(x)$ is the solution found by the current hypothesis. However, the bias-variance concept assumes that the models exist in the same hypothesis space, due to this, this formula cannot be used to compare models of different types.

2.3.3 Robustness

Predictive models are developed and trained to recognize data that comes from the same distribution as the data they had been trained on, such that they can provide a performance similar to that obtained during training. A robust predictive model can maintain the same level of performance even when small perturbation is applied to the data (noise

perturbation) or to the model internal parameters (parameter perturbation). Noise perturbation and parameters perturbation can be explained through the following equations (Jin and Branke (2005)), respectively:

$$\hat{f}(x) = \frac{1}{N}\Sigma_{i=1}^{N}[f(x) + noise_i] \tag{2.27}$$

$$\hat{f}(x) = \frac{1}{N}\Sigma_{i=1}^{N} f(x + \delta_i) \tag{2.28}$$

Equation 2.27 calculate the expected fitness function with noise perturbation, where $\hat{f}(x)$ is an approximation of the function $f(x)$ and $noise_i$ is the added noise. On the other hand, equation 2.28 calculates the expected fitness function (using Monte Carlo integration) with parameter perturbation where δ_i represents the perturbation in the system parameters. It can be noted that the noise in equation 2.27 is an independent variable that is added to the fitness function, while in equation 2.28 the uncertainty is part of the design variables. Due to this, in the parameter disturbance case, even if the noise was normally distributed, the expected fitness function depends on the shape of the function $f(x)$ at point x. The internal parameters of the predictive models may be subjected to noise due to: noisy observations, estimating the parameters from a finite number of samples, and over-simplification of the problem parameters.

Finding robust predictive models have been investigated in the field of decision (Bertsimas and Thiele (2006)) as well as machine learning (Caramanis et al. (2011)). Early works which address parameter perturbation in decision making use stochastic programming. In this approach, the uncertainty was treated as a random variable with known distribution probability (Kali and Wallace (1994) and Birge and Louveaux (2011)). Nevertheless, knowing the actual distribution of disturbances is rarely possible in real world problems. More recent approach which addresses parameter perturbation is the robust optimisation(Bertsimas and Sim (2003) and Bertsimas and Sim (2004)). It assumes that the distribution of the disturbances is known within certain bounds. In machine learning, the task of finding a classification or regression model can be considered as finding the optimal boundary with respect to unknown probability distribution which can be approximated using a finite set of samples (Xu et al. (2009)).

In the following subsection stability, a more restrictive concept than robustness, is discussed.

2.3.3.1 Stability

Generally, there are two main differences between stability and robustness (Jen (2003)). Robustness can be applied to wider classes of systems, perturbation and features of interest in the system. In addition, robustness provide a more general framework to investigate the system behaviour in: dynamically changing environment, the evolvability of the system over time, the cost and benefits of robustness and the effect of different perturbation on different features of the system.
On the other hand, a system is said to be stable if its performance is not affected by slight modification in the input data. Several approaches were proposed to create predictive models that are less sensitive to perturbation. An example is the uniform convergence of empirical quantities to their means (Vapnik and Kotz (1982)), where a bound on the generalisation error can be obtained for a large number of empirical risk minimization algorithms, such as, VC-dimension and fat-shattering dimension. Nevertheless, this approach cannot be applied with machine learning algorithms that have unbounded search space (like K-nearest neighbour). In (Bousquet and Elisseeff (2002)) a new definition for stability was proposed, where the notion of uniform hypothesis stability is introduced. In this approach the focus is on how the algorithm searches the space rather than the actual size of the space. An example of this approach is the use of regularization methods (Bousquet and Elisseeff (2002)).

2.3.4 Adaptation

The performances of predictive systems that are developed in a static environment often deteriorate over time when applied online. To solve this problem and to improve the performance of predictive systems in dynamically changing environments, adaptation is introduced in the design of predictive systems. Adaptation can have a crucial effect on the predictive system accuracy as well as the amount of resources used.
In order to build an adaptive predictive system, the first task is to identify when the adaptation will take place. However, rather than having a mechanism that identifies the need for adaptation, many adaptive systems use a periodical adaptation mechanism regardless of the actual need (Kadlec et al. (2011)). Continuous adaptation can result in a waste of resources.
System adaptability can be evaluated with respect to changes in the predictive system

accuracy. Furthermore, the need for adaptation can be assessed by measuring the cost of adaptation. If the benefits acquired from the adaptation are greater than the drawbacks then the adaptation is performed. According to (Žliobaitė et al. (2015)) the cost of the adaptation can be decomposed into four components:

- Computational cost: is a function of the processing power and memory consumption.
- Opportunity cost: is the lost cost, it has a non-zero value if the system is unable to deliver prediction during an adaptation.
- Label cost: is the cost of obtaining labels.
- Communication cost: is the cost of transmitting the data.

Once the need for adaptation is triggered the actual adaptation procedure is performed. Generally, adaptation can be carried out at two levels: low level where the system parameters are updated or higher level where the structure of the system is updated. In both cases the main goal of the resultant system is to be able to adapt to slow changes and drifting in the measurements as well as to sudden changes. In general, the adaptation of the predictive model can follow one of the following four strategies (Žliobaitė et al. (2015)):

- Fully incremental: the update is performed using the previous model and the latest instance.
- Summary incremental: the update is performed using the previous model, data summary and the latest instance.
- Batch incremental: the update is performed using a fixed number of instances stored in a buffer.
- Non-incremental: the model is re-built at every adaptation.

2.3.5 Transparency

Transparency means that a model, its parameters value, equations and assumptions can be easily interpreted by experts as well as non-experts. Such model provides the non-expert with a general idea about the workflow of the model. Also it enables the expert to evaluate the model performance. Transparency is a subjective and vague term as only the user can judge the transparency of a model. Nevertheless, certain methods were proposed to measure the interpretability of models, such as, the minimum

description length principle for rule-based classification systems (Nauck (2003)).
Most predictive systems are viewed as black boxes for non-expert users, where input goes in and prediction comes out without any explanation of the procedure used to obtain the results. It was stated in (Nauck (2005)) that model transparency can be useful in the following scenarios: to support human decisions in applications that depend on such type of decisions, to check for modification in knowledge when it is used as a prior knowledge in data analysis and to explain the obtained solution for non-experts.
An important thing to note is the difference between an explanatory model and a transparent predictive model (Nauck (2005)). Explanatory models are designed to describe the data to the user (they use all the available data and do not care for the generalisation) and they are not used to provide prediction. Also they require an immediate result to be provided to the user. Meanwhile, transparent predictive models are models that can be easily interpreted by the user and they are developed to provide prediction for a certain process, where their generalisation ability is a key factor. In addition, the model must be available when the first prediction is due. Increasing model transparency often results in reducing model accuracy (Nauck (2003)). In certain application where a transparent model is required and a certain level of accuracy is needed, a trade-off between the two criteria has to be maintained.

2.4 Predictive System Optimisation

Depending on the number of criteria used to evaluate the performance of the predictive system and the importance associated with each criterion, one of the following approaches can be used to optimise the performance of the predictive system. In the following subsections four of the most common optimisation approaches are described for optimising single as well as multiple criteria:

2.4.1 Single Objective Optimisation

In this approach the predictive system performance is optimised using a single criterion. Traditionally, the accuracy of the prediction is used as the main optimisation criterion. In the case where other criteria are also considered in the optimisation, these criteria can be presented as constraints in the optimisation process. For example, the model complexity

can be presented as a constraint to ensure that the developed model does not overfit or underfit the training data and that it generalises well on unseen data. An example of this approach is the use of cross-validation techniques (Anguera et al. (2007)). Figure 2.9 shows a simple illustration of this approach, where Cr represent a single criterion.

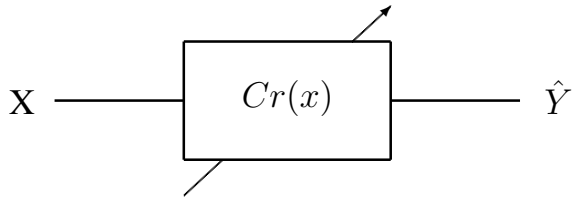


Figure 2.9: *Optimising a single criterion*

2.4.2 Scalarized Multi-Objective Optimisation

In order to optimise the performance of a predictive system using multiple criteria, one approach is to combine these criteria using a scalarization function. Figure 2.10 provides a simple illustration of this approach, where $w_1, , w_k$ are the weights associated with the k criteria $Cr_1(x), , Cr_k(x)$.

An example of this approach is optimising the accuracy with respect to the model com-

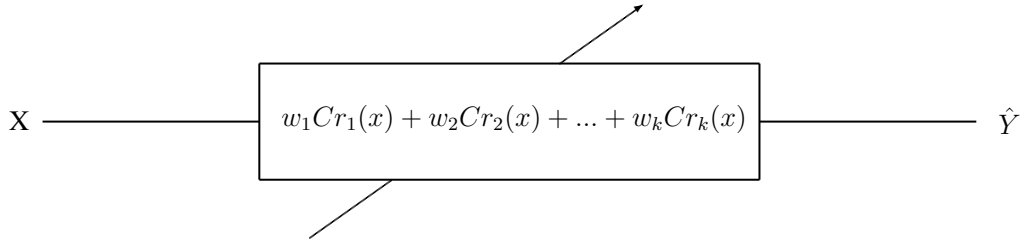


Figure 2.10: *Optimising multiple criteria using scalarized MOO*

plexity, which can be achieved using a number of methods, such as, using regularisation terms to penalize models with high complexity (Barron (1991)), the function for this approach is given below Jin and Sendhoff (2008):

$$\hat{f}(x) = \epsilon + \lambda\Omega \tag{2.29}$$

where ϵ is the prediction error, λ is a hyperparameter and is chosen in the range $(0 < \lambda < 1)$ and Ω is the model complexity.
Furthermore, there are algorithms that optimise more than one objective as part of their training procedure such as the SVM which optimises both accuracy and complexity through the use of SRM framework (Cortes and Vapnik (1995)).
Another example for scalarized optimisation is the weighted formula (Andersson (2000)), where a certain weight value is associated with each criterion; the weights represent the degree of importance each criterion have. Depending on the value of the criteria and the provided weights a certain model is preferred over the rest. The mathematical representation of the weighted formula is shown below:

$$\hat{f}(x) = \sum_{i=1}^{N} w_i Cr_i(x) \tag{2.30}$$

2.4.3 Multi-Objective Optimisation

Multi-objective optimisation finds a set of non-dominated models (solutions) that trade-off a set of conflicting criteria. The general mathematical formula of MOO can be expressed using the following constraint optimisation problem (Deb (2001)):

$$\begin{array}{lccc} \min(\max) & Cr_k(x) & k = 1, 2, .., K \\ Subject \quad to & g_j \geq 0 & j = 1, 2, ..., J \\ & h_m = 0 & m = 1, 2, ..., M \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1, 2, ..., N \end{array} \tag{2.31}$$

A solution that satisfies the equality constraints h_m as well as the inequality constraints g_j is called a feasible solution, while the solution that does not satisfy these constraints is called an infeasible solution. The input x_i is limited by a lower bound $x_i^{(L)}$ and an upper bound $\leq x_{(i)}^{U}$.
Since there is typically no single solution that can optimise all of the conflicting criteria, most MOO algorithms aim to find a set of non-dominated Pareto-optimal solutions. These solutions (models) are non-dominated with respect to each other and they represent a trade-off between the different criteria.
According to Deb (2001) two conditions must be satisfied for solution (A) to dominate solution (B): solution (A) must not be worse than solution (B) in any criterion, and solu-

tion (A) must be better than solution (B) in at least one criterion.
The most common types of Pareto-based Multi-Objective Optimisation are evolutionary based MOO such as Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al. (2002)), Multi Objective Particle Swarm Optimisation (MOPSO) (Parsopoulos and Vrahatis (2002)) and Strength Pareto Evolutionary Algorithm 2 (SPEA-2) (Zitzler et al. (2001)) among others.

2.4.4 Hierarchical Optimisation

In this approach a priority vector (ranking) is associated with the criteria used in the optimisation process. The optimisation process starts with the criterion that has the highest level of importance (highest rank) and gradually optimises the rest of the criteria in a descending order according to their rank. Figure 2.11 illustrate the hierarchical optimisation approach.

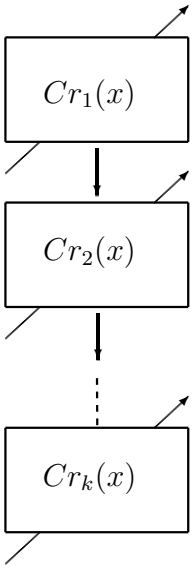


Figure 2.11: *Optimising multiple criteria using hierarchical optimisation*

An example of this approach is the lexicographic method, where a decision maker provides a ranking vector and according to this vector the criteria are optimised sequentially from the highest rank to the lowest rank. After each round of optimisation the resultant models are compared according to the current major criterion. If a set of models have comparable values with respect to that criterion, further comparison is held according to the next lower rank criterion. However, defining the rank in which the criteria are opti-

mised needs predefined information about their level of importance. Furthermore, once a dominated model with respect to a certain criterion is found, the comparison ends and the remaining lower rank criteria are not considered in the comparison (Rentmeesters et al. (1996)).
The mathematical representation of the lexicographic method can be given as (Jee et al. (2007)):

$$\begin{aligned} &\min Cr_i(x) \\ &Subject \quad to \qquad Cr_j(x) \leq Cr_j(x_j^*) \\ &i = 1, 2, , K, \quad j = 1, 2, , i-1 \quad if \quad i > 1 \end{aligned} \tag{2.32}$$

Where $Cr_i(x)$ is the current criterion to be optimised, $i = 1$ for the highest important criterion and as the optimisation proceed $(i > 1)$ the previous optimisation criteria are converted into inequality constraints. The $Cr_j(x_j^*)$ represent the optimal value given by a prior attained solution $x_j^* = \arg\min Cr_j(x)$ subjected to constraints from the previous levels. Thus as the final level of the optimisation process is reached, the number of constraints increase to $K - 1$.

2.4.5 Comparing Optimization Approaches

In the literature many studies compare the effectiveness of the different approaches of MOO, focusing on the advantages and drawbacks for each approach. An example is the work presented in (Jin and Sendhoff (2008) and Freitas (2004a)), where Single Objective Optimisation (SOO), weighted formula (scalarized MOO) approach and Pareto based MOO approach are compared based on the quality of the models these approaches generate. The following section provides a brief comparison between these three approaches.
Starting with the weighted formula, the main advantage for this method is its simplicity and ease of use where all the criteria can be combined in a single weighted function. However, it suffers from a number of drawbacks (Freitas (2004b)). One of the main drawbacks of the scalarized MOO is the need to define additional information, like providing the weights vector for the weighted sum method. Another drawback is that each criterion is measured using different units and they often have different scales, combining them in a single equation results in mixing different types of measurements. Furthermore, normalising the criteria defines a single scale, but it can result in losing the

sense of how good or bad the criterion value is.

On the other hand, Pareto-based approach requires all of the criteria to be measured independently, while this might not always be the case in the scalarized MOO approach. Measuring the different criteria independently is not always feasible. For example, the model complexity of an SVM model cannot be measured independently of its accuracy, as both criteria are optimised together during the training phase. However, these criteria can be measured for many predictive model types, such as, decision trees. Though this approach is more complex, it does not suffer from the randomness found in the scalarized MOO approach which comes from defining the parameters in an ad-hoc manner (Freitas (2004b)). It can be argued that running a scalarized MOO multiple times can provide a set of solutions that lies on the Pareto front and by that the unnecessary complexity of the Pareto-based approach can be avoided. However, this claim ignores the following arguments: running a scalarized MOO method multiple times is an ad-hoc approach, hence each time different set of weights are used and the results obtained from the previous run are not considered. Also, there is no clear method for determining the number of iterations for the algorithm to find all the non-dominated solutions. Furthermore, the solutions found might not capture the actual distribution of solutions on the Pareto front which can result in losing the diversity of the generated models. Finally scalarized MOO cannot find solutions in non-convex Pareto-front.

Although the scalarized MOO returns a single solution while the Pareto-based MOO returns a set of solutions, the latter approach can be especially useful when the output of the predictive system needs to offer a range of solutions (models) to an experts (a decision maker) to choose from according to their preference. In practice, identifying the Pareto front for a real world problem cannot be achieved accurately, but it is loosely assumed that the solutions obtained by Pareto based algorithms are Pareto optimal solutions.

2.5 Summary

The learning process in predictive systems can be decomposed into three components: representation, evaluation and optimisation of the predictive system. This chapter explores these components to establish the theoretical background for the work presented

in this thesis. Since this thesis employs multiple-criteria in the optimisation process for complex learning prediction systems, this chapter defines predictive systems architectures, the criteria used in their evaluation and the optimisation approaches used to generate the predictive system/model.

This chapter started with defining the architectures of the predictive systems in Section 2.2, from single predictor to complex pool of competing predictors. Then Section 2.3 examines the multiple criteria used in evaluating predictive systems performance. The criteria considered in this Section include: the accuracy, model and algorithmic complexity, robustness, adaptation and transparency. In section 2.4 many optimisation approaches used to optimise the predictive system performance with respect to the evaluation criteria are explained. This Section considers the main approaches used to optimise a single as well as multiple criteria.

The next Chapter presents a general design cycle of the predictive system which focuses on the generation and optimisation of the base predictors using multiple criteria. Furthermore, it conducts a new experiment which compares two MOO approaches and highlights the cost and benefits obtained from including multiple criteria in the optimisation of machine learning models in different classification settings. The optimisation approaches used are: scalarized MOO and the Pareto-front MOO.

Chapter 3

Design Cycle of Multi-Component, Multi-Layer Predictive System and Base Models Generation

3.1 Introduction

This chapter introduces the design cycle of the MCMLPS considered in this thesis and focuses on the generation and optimisation of the base predictors using multiple criteria. This design cycle (discussed in Section 3.2) focuses on certain aspects of MCMLPS, such as, local versus global models, evaluation of base predictors using single or multiple criteria and the optimisation approaches used for this system. Furthermore, a case study for evaluating and optimising the performance of the base models for MCMLPS is introduced in Section 3.3. This case study follows two optimisation approaches to optimise the performance of the base models using multiple criteria. It provides a critical evaluation of the literature reviewed in the previous chapter. In addition, it helps in setting the experimental framework of this thesis by highlighting the benefits and drawbacks of using certain criteria and optimisation approaches in evaluating and optimising the performance of the predictive system.

3.2 General Design Cycle of MCMLPS

The design cycle of MCMLP is shown in Figure 3.1. This work does not investigate the process of data acquisition as it assumes the data has been already collected. The proposed design cycle encompasses the following stages:

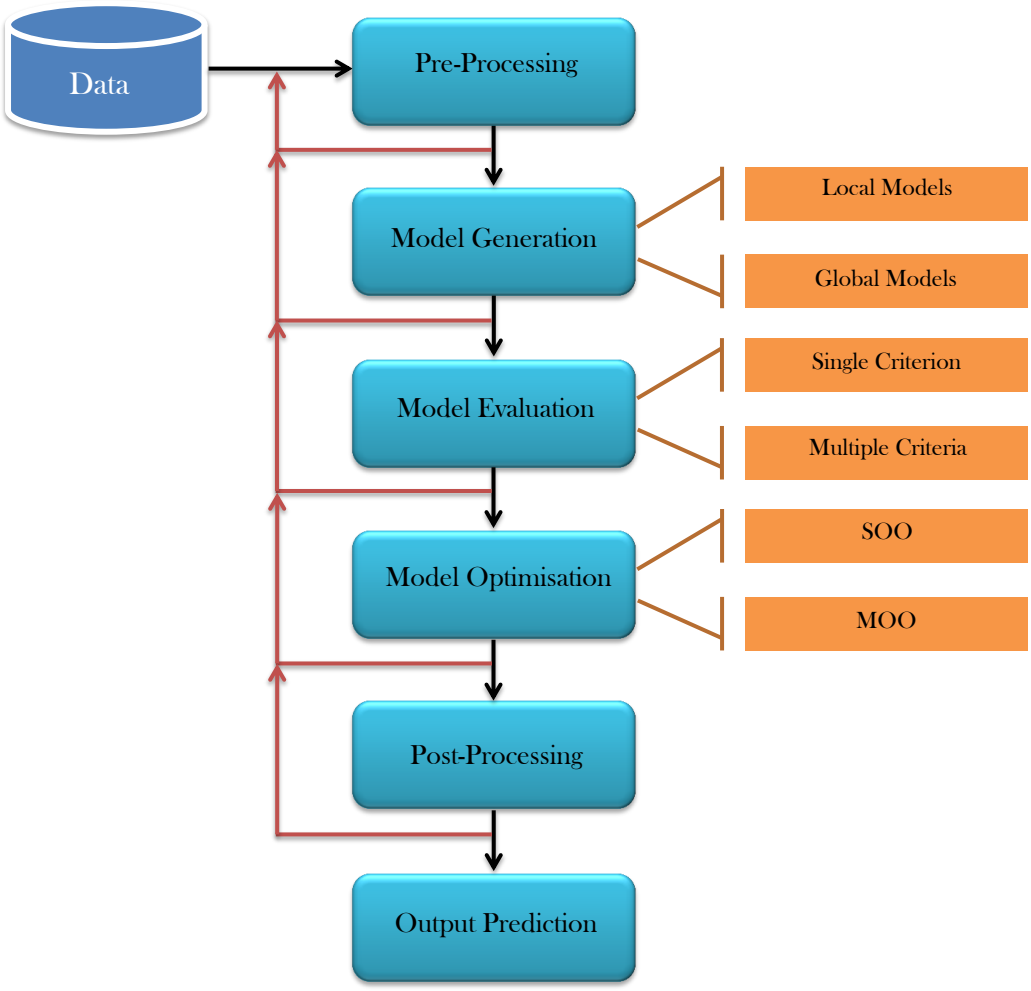


Figure 3.1: *Generalized design Cycle of MCMLPS*

3.2.1 Pre-Processing

In real world problems, the collected data can be affected by one or more of the following complications: noise, missing values, inconsistent data and high dimensionality. Thus, pre-processing the data before training the predictive models can improve the quality of the prediction and/or reduce the time required to deliver the prediction (Han et al. (2011), Bruha and Famili (2000), Salvador et al. (2016) and Zliobaite and Gabrys (2014)). A single or multiple stages of pre-processing methods can be applied to the data. In general, pre-processing techniques can be used to perform different tasks. These tasks include (but not limited to) the following:

- Outlier removal: An outlier is defined as a point that lies outside the mean distribution of the input data (Theodoridis and Koutroumbas (2006)). The existence of outliers can be a result of noisy measurements. In most cases outliers are considered as noise or exceptions and are discarded during the pre-processing stages. However, when the number of outliers is large or when they represent an event of interest to the designer (for examples, the outliers in fraud detection, where rare cases of frauds are more important than regular events), outliers can be analysed through outliers mining techniques (Han et al. (2011)).
- Data normalization: The ranges of the data features can vary widely as they represent different aspects of the prediction problem. For example, in customers data age and salary can have completely different ranges which do not necessarily represent their influence in the prediction problem. In machine learning methods, especially if these methods provide their prediction using a distance based metric, such as, nearest neighbour or clustering methods having large differences in the scales of the features can affect the accuracy of the prediction. Thus often normalising the data can lead to improving the performance of the prediction method (Han et al. (2011)).
- Missing values: In practise, some of the data features can have missing values for certain instances. This can be due to one or more for the following: malfunction in the equipment used to collect the data, missing information for certain users, error in entering the data, etc (Mitchell (1998) and Alpaydin (2014)). Pre-processing techniques can be used to handle missing values problem. If the data set is large enough, the data instances with missing values can be discarded, however, in real world problems this is seldom the case. The missing values can be predicted using

one of the following techniques (Han et al. (2011)): filling the values manually, replacing missing value with a global constant, using the mean of the respective feature to fill in missing values or predicting missing values.

A number of machine learning approaches have been proposed to handle missing values without explicitly provide alternative values for them. Examples on these methods can be found in neural network ensembles such as the network reduction method proposed by Sharpe and Solly (1995) which trains a set of Multilayer Perceptrons (MLPs) on a different possible combination of the features. Furthermore, Krause and Polikar (2003) developed a NN ensemble which deals with missing values by training its base predictors on random subsets of the features. Another approach is proposed by Juszczak and Duin (2004), where an ensemble consists of one class classifiers, each trained on a single feature. Therefore, if a feature has a missing value for certain sample, the classifier can still provide prediction for this sample using the other features.

On the other hand, some of the well-known methods used to generate decision trees have additional mechanisms that allow them to deal with missing values, such as, ID3 (Quinlan (1986)) where an additional edge in the tree is provided for each missing feature. This edge contains the possible values for the missing feature. An extension of ID3 is the C4.5 (Quinlan (2014)) which uses probabilistic approaches to deal with missing values.

- Dimensionality reduction: One of the main factors that control the complexity of the predictive model is the size and dimensionality of the data. Thus reducing the dimensionality of the data can reduce the complexity of the model. Furthermore, using smaller number of features (without loss of information) can better explain the underlying process which generates the data as well as help to visualize and analyse the data (Alpaydin (2014)).

  Dimensionality reduction can be achieved using one of the following two approaches:

  - Feature selection: In this approach a subset of the features that contains the most information about the prediction problem is selected. The subset is selected based on a predefined metric, such as, features correlation (Tumer and Ghosh (1996)) or mutual information (Cover and Thomas (2012)).
  - Feature extraction: In this approach the original features are combined into a

new set that has a fewer number of features. Feature extraction methods can be classified according to the type of learning method into: supervised and unsupervised methods.

The most widely used feature extraction methods are Principle Component Analysis (PCA) and Linear Discriminate Analysis (LDA) which are linear projection methods that can be applied to supervised as well as unsupervised methods. Examples on non-linear dimensionality reduction methods are isometric feature mapping (Tenenbaum et al. (2000)) and locally linear embedding (Saul and Roweis (2000)).

3.2.2 Model generation

In this stage, the processed data is used to train the base models of MCMLPS. The training data is usually limited and expensive to obtain (Budka (2010)). Furthermore, to avoid overfitting, not all of the available data is used in training the base models. Some of the available data should be reserved for testing the performance of the generated models. Normally, the more data is used in training the model, the more the model represents the problem and the better is its performance (Ruta (2003)).

In ensemble learning, the diversity of the base models plays an important role in determining the overall performance of the system. Thus the base models of such systems are often trained on different versions of the available data. The models can be trained either on a slightly different replicas of the original data, like in Bagging (Breiman (1996)), where each base predictor is trained on a bootstrapped sample of the original data, or they can be trained on modified versions of the original data, like in Boosting (Freund and Schapire (1996)), where each new base model is trained on a new weighted version of the data set. Such models are known as global models as they are trained on all of the available data. On the other hand, a model that is trained on a local subset of the features and/or the data, is known as a local model. Local models can be trained on disjoint or intersected subsets of the data (Rodriguez et al. (2006) and Kadlec and Gabrys (2011)).

3.2.3 Model evaluation

Historically, the generalisation ability of the predictive system is used as the main criterion in evaluating the performance of the predictive models. However, as it has been

discussed in Chapter 2, depending on the prediction problem and the user preference, the performance of the predictive systems can be evaluated using single as well as multiple criteria. The criteria can be measured using different methods; some of these measures are specified to only one type/class of predictive models, while other measures can be used across different models. A universal measure is a measure that can be used to evaluate a certain criterion across all types of predictive models. However, these measures can be defined for some but not all of the criteria discussed in the previous Chapter. The feasibility of defining universal measures for these criteria is explored below.

A universal measure can be defined for the accuracy of the predictive model, where the test error of the prediction, can be estimated and compared across different types of predictive models. On the other hand, a more difficult task is to define a universal measure for the model complexity. Different predictive models have different structures. Depending on these structures, the complexity of the models can be evaluated. A number of methods for measuring the model complexity were discussed in Chapter 2. Furthermore, the algorithmic complexity can be measured using the asymptotic analysis and the big O notation. Nevertheless, these measures do not take into consideration the programming language used or the machine limitation. Due to this, the algorithmic complexity is often captured through monitoring the execution time and the memory usage.

Meanwhile, many methods were proposed to measure the diversity among the base models of MCMLPS. However, there is no universal measure or notion of diversity (Kuncheva and Whitaker (2003), Brown and Kuncheva (2010) and Bi (2012)).

Furthermore, the adaptation ability of predictive models depend on the type of the model and the operating environment. The need for adaptation can be assessed by measuring the cost of adapting. If the benefits acquired from the adaptation are greater than the losses, then the adaptation is performed (Žliobaitė et al. (2015)). The cost of adaptation can be used as an additional criterion or a constraint in the designing process of MCMLPS.

In addition, considering the robustness and stability of the developed models can help in reducing the model sensitivity to small perturbations in the environment and/or in the models parameters (Xu et al. (2009)). However, the approaches used to ensure the robustness and stability of the developed model depend on the type of the model used and the optimisation process.

3.2.4 Model optimisation

In this stage, the evaluation criteria are used to optimise the performance of the predictive system. As has been discussed in Chapter 2 (Section 2.4), depending on the number of the criteria included in the optimisation process, a single or multiple objective optimisation approaches can be followed.
The optimisation process can be performed either in one iteration (such as optimising accuracy and complexity using regularization term Barron (1991)) or repeated multiple times (such as optimising multiple criteria using NSGA-II Deb et al. (2002)) until the required performance is achieved. In methods like Boosting, the optimisation process is repeated each time a new classifier is added to the ensemble. The number of iterations for the optimisation process also depends on the availability of the computational resources.

3.2.5 Post-processing

Post-processing can be applied to the predictive models as well as their output prediction (Bruha and Famili (2000)). A subset of or all the trained models can be used as the base predictors for the MCMLPS. The selection of the models included in the system can be made using a single or multiple criteria. These criteria are not necessarily the same as the criteria used in evaluating the performance of the base models. For example, the selection can be made purely based on the accuracy, while the models are evaluated using both accuracy and model complexity. Furthermore, a given characteristic of the models can be used as a selection criterion, such as, using a measure of locality or diversity among the models. Common examples of selecting the models based on their accuracy are (Ruta (2003)): selecting the top N scoring models, removing the worst $N\%$ models or choosing models that perform better than random guessing.
Once the models are selected, a combiner is used to provide the final prediction of the system. Examples of combiner methods are (reviewed in Ruta and Gabrys (2000)): Majority vote, plurality vote, averaging combiners and their weighted versions.
Post-processing can also be applied to the extracted knowledge, such as performing a post-pruning to a decision tree or rule truncation (Toivonen et al. (1995)) for decision rules. In addition, the extracted knowledge can be interpreted and explained through visualization or documentation (Bruha and Famili (2000)).

The next Section investigates the generation of predictive models using multiple criteria. The main results of this study has been published in (Al-Jubouri and Gabrys (2014)). The aim of the study proposed in the next Section is to explore the feasibility of including multiple criteria in the design process of the base components of MCMLPS. Furthermore, it compares different multi-objective optimisation approaches and examines the main advantages and drawbacks of evaluating and optimising the performance of the base predictors using multiple criteria.

3.3 Comparing Multi-criteria Predictive Models generated from Scalarized MOO and Pareto-based MOO

This Section investigates the evaluation of MCMLPS base models using multiple criteria. The criteria included are: predictive model accuracy, model complexity, and algorithmic complexity (related to the learning algorithm and prediction delivery) captured by monitoring the execution time. Furthermore, this study compares and analyses the predictive models resulted from two optimisation approaches. The first approach is a scalarized MOO, where the models are generated from optimising a single cost function that combines the criteria. On the other hand, the second approach uses a Pareto-based MOO to trade-off the three criteria and to generate a set of non-dominated models. The first

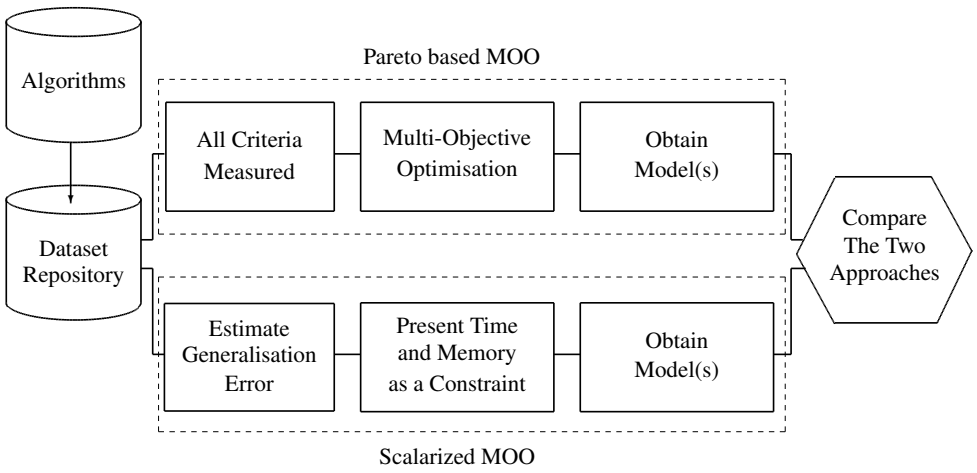


Figure 3.2: *The two approaches for evaluating and optimising predictive models.*

approach estimates the generalisation error of the predictive models by optimising the

accuracy with respect to the complexity and then the execution time and memory usage is used as a hard constraint to choose the most appropriate model.

On the other hand, the second approach requires all three criteria (accuracy, model complexity and algorithmic complexity) of the predictive models to be measured separately. Then the performance of the predictive models is optimised using a Pareto-based MOO technique and the resultant non-dominated models are presented to a decision maker to choose from. This study shows that the models generated from Pareto-based MOO approach can be more accurate and more diverse than the models generated from scalarized MOO approach. Figure 3.2 illustrates the two approaches for evaluating and optimising predictive models.

The data sets used in this experiment are shown in Table 3.1. The aim of this experiment is mainly to investigate the optimisation of predictive models using multiple criteria. Furthermore, it aims to compare between different optimization approaches, with no intention to prove that one approach is superior to another. Thus the data sets have been chosen such that they have a varying number of samples and generally a small number of features. The varying sizes of the data sets helped in studying the effect of the data size on optimizing the three considered criteria using the two optimization approaches. On the other hand, having a small number of features helped in interpreting the results. Furthermore, the effect of increasing the number of features in the data on the accuracy of the prediction is studied using the Gaussian data sets, where the same data set is described using 2, 4 and 8 features. Descriptions of the data sets used in this work are presented in Appendix A.

Table 3.1: *Data set details*

Number	Data sets	No. of examples	No.of features	No. of classes
1	Cloud	5000	2	2
2	Concentric	2500	2	2
3	Cone-Torus	400	2	3
4	Gaussian 2D	5000	2	2
5	Gaussian 4D	5000	4	2
6	Gaussian 8D	5000	8	2
7	Shuttle	58000	9	7
8	Synthetic	1250	2	2

3.3.1 Methodology

In order to compare the predictive models generated using the two optimisation approaches discussed above; measures for the criteria included in the optimisation should be defined. The criteria included are accuracy, model complexity and algorithmic complexity. Here, a universal measure can be defined for the predictive models accuracy (such as, the mean square error). Meanwhile, the algorithmic complexity can be captured through measuring the execution time and the memory usage required by the training algorithm. If only one type of models is used (that has the same structure) then only the execution time is considered since the memory usage will be the same for all models.
Both accuracy and execution time can be measured and compared across predictive models from different families. However, as mentioned before, this is not the case for the model complexity. Different models require different complexity measures. Due to this, predictive models from different families cannot be compared based on their complexities. In order to allow the evaluation and optimisation of the predictive models using all three criteria, this study uses only one type of predictive models. A feed forward NN with one hidden layer and ten hidden units has been used. The output of this network is defined as:

$$\hat{f}(x_t, w^U, w^H) = \sum_{j=1}^{m} w_j^U \phi(\sum_{i=0}^{n} w_{i,j}^H x_{it}) \tag{3.1}$$

U and H are the weights of the output and hidden layers respectively. In the Pareto-based MOO approach the neural network is trained using Levenberg-Marquardt algorithm (Marquardt (1963) and Hagan and Menhaj (1994)), while in the scalarized MOO approach the neural network is trained using Bayesian regulation backpropagation algorithm (MacKay (1992) and Dan Foresee and Hagan (1997)). The transfer functions for the hidden layers and the output layer are hyperbolic tangent and linear functions respectively.
In both approaches the accuracy is measured using the mean square error. Meanwhile, the complexity of the developed models is bounded using the sum of the square weights, as shown in equation 4.1 (Jin and Sendhoff (2008)).

$$\Omega = \sum_{i=1}^{M} w_i^2 \tag{3.2}$$

The weights used in this experiment include: the input layer weights and hidden layer weights.
Due to the use of only one type of NNs, the memory usage is fixed for all the networks and the execution time T (measured in seconds) is used to capture the algorithmic complexity of the predictive models.

The Pareto-based MOO technique used in this study is NSGA-II (Deb et al. (2002)). The NSGA-II unconstrained optimisation equation for the NN is given below:

$$\begin{array}{lll} Min & (Cr_1, Cr_2, Cr_3) & \\ where & & \\ Cr_1 & = & Error \\ Cr_2 & = & \Omega \\ Cr_3 & = & T \end{array} \tag{3.3}$$

The parameter setting for the NSGA-II are:

- Population size: 10
- Selection function: Tournament selection
- Crossover fraction: 0.8
- Crossover function: Single point crossover
- Mutation rate: 0.05
- Mutation function: Uniform
- Termination condition: exceed 10 iterations.

On the other hand, in the scalarized MOO approach, the NN is trained using a Bayesian regulation, where the square error and the sum of the network weights are combined in a single cost function and the training algorithm aim to minimize this function. In this approach time can be presented as a hard constraint to choose the final model.

3.3.2 Results

The scalarized MOO and the Pareto-based MOO approaches were applied to the eight data sets shown in Table 3.1. Figure 3.3 compares the non-dominated predictive models generated from the Pareto-based MOO approach and the models generated from multiple runs of the scalarized based MOO approach. Due to the different scales of the three criteria, a 3D illustration is not suitable to compare the results. Furthermore, using 2D representation to compare two criteria at a time does not allow an accurate comparison, as the Pareto-based MOO approach optimises all the three criteria simultaneously while the second approach optimises only two criteria and presents the third one as a constraint. Due to this, each criterion is compared individually for the two approaches. Moreover, a kernel smoothing function (presented in Bowman and Azzalini (1997)) is used to represent the results for the three criteria.

The first column of Figure 3.3 shows that the error of the Pareto-based MOO approach has a lower starting point than the error of the scalarized MOO approach, this means that, in this experimental work, the model generated from the Pareto-based MOO approach can often have a lower error value than that generated from scalarized MOO approach. Furthermore, the highest peak of the Pareto-based MOO approach is centred on a lower error value than the scalarized MOO approach. This indicates that most of the models generated from the Pareto-based MOO approach have a lower error value than those generated from the scalarized MOO approach. On the other hand, the model complexity of the two approaches had varied widely across the eight data sets. However, the model with the highest complexity is often found by the Pareto-based MOO approach. Furthermore, both approaches have produce models with high accuracy when applied to a large multi class data set like the shuttle data set (Figure 3.3(s)). Also, increasing the dimension of the Gaussian data sets (2D, 4D and 8D) did improve the accuracy of the models generated from both approaches (as can be seen in Figure 3.3 (j),(k) and (i) respectively).

In addition, it can be noticed from Figures 3.3 (o) and 3.3 (u) that the complexities of the models generated from scalarized MOO approach for the two data sets (Gaussian 8D and Shuttle) have a single peak on a low value, while the complexities of the Pareto-based MOO models continue to large values (up to 5×10^7). This is due to a large complexity value for one or more of the non-dominated models found on the Pareto front of the MOO approach.

Finally, the execution time of the scalarized MOO models is almost fixed while the

Pareto-based MOO models have a varying execution time for the eight data sets, as shown in the third column of Figure 3.3. Generally, the time required by the NSGA-II to find the non-dominated models is large. This is due to the high algorithmic complexity of the NSGA-II algorithm, which is $O(MN^2)$ (Deb et al. (2002)), where N is the population size and M is the number of objectives, with an algorithmic complexity of the individual chromosomes (NN trained with Levenberg-Marquardt algorithm) equals to $O(C^2)$, where C is the number of examples (Zhou and Si (1998)). On the other hand, the algorithmic complexity of the second approach is $O(C^2)$.

Pareto-based MOO techniques produce a set of non-dominated models, and the final model is selected from them. Often this model is selected based on the knowledge of an expert (a decision maker). In order to automate the choice of the mode, different techniques can be used (Deb (2003)). These techniques are applied either after the non-dominated models are found (post-optimal techniques) or during the optimisation process. One of the post-optimal techniques is to measure the distance between the models and a reference point, which is an imaginary point that has the minimum values (founded by any of the non-dominated models) of all the criteria. On the other hand, the execution time can be used as selection criteria in the scalarized MOO approach.

3.3.3 Increasing the Population Size and the Maximum Number of Generation in the Pareto based MOO

This section studies the increase in population size and the maximum number of generation for the Pareto based MOO. The results showed that, this can improve the accuracy of the predictive models, as can be seen in the first column of Figures 3.4.

However, running the Pareto-based approach for a very long time and increasing the population size, results in increasing the upper bound of the model complexity. It can be noticed from Figures 3.4 (m), 3.4 (o), 3.4 (v) and 3.4 (u) that the complexities of the models generated from scalarized MOO have a single peak on a low value. Meanwhile, the complexities of the Pareto-based MOO models have a varied distribution over the x-axis scale and can reach large values. As has been indicted in the previous Section, this is due to having large complexity value for one or more of the non-dominated models found on the Pareto front. Nevertheless, for the same data sets there are non-dominated models that have a lower complexity value than the scalarized approach with different trade-off between complexity and execution time.

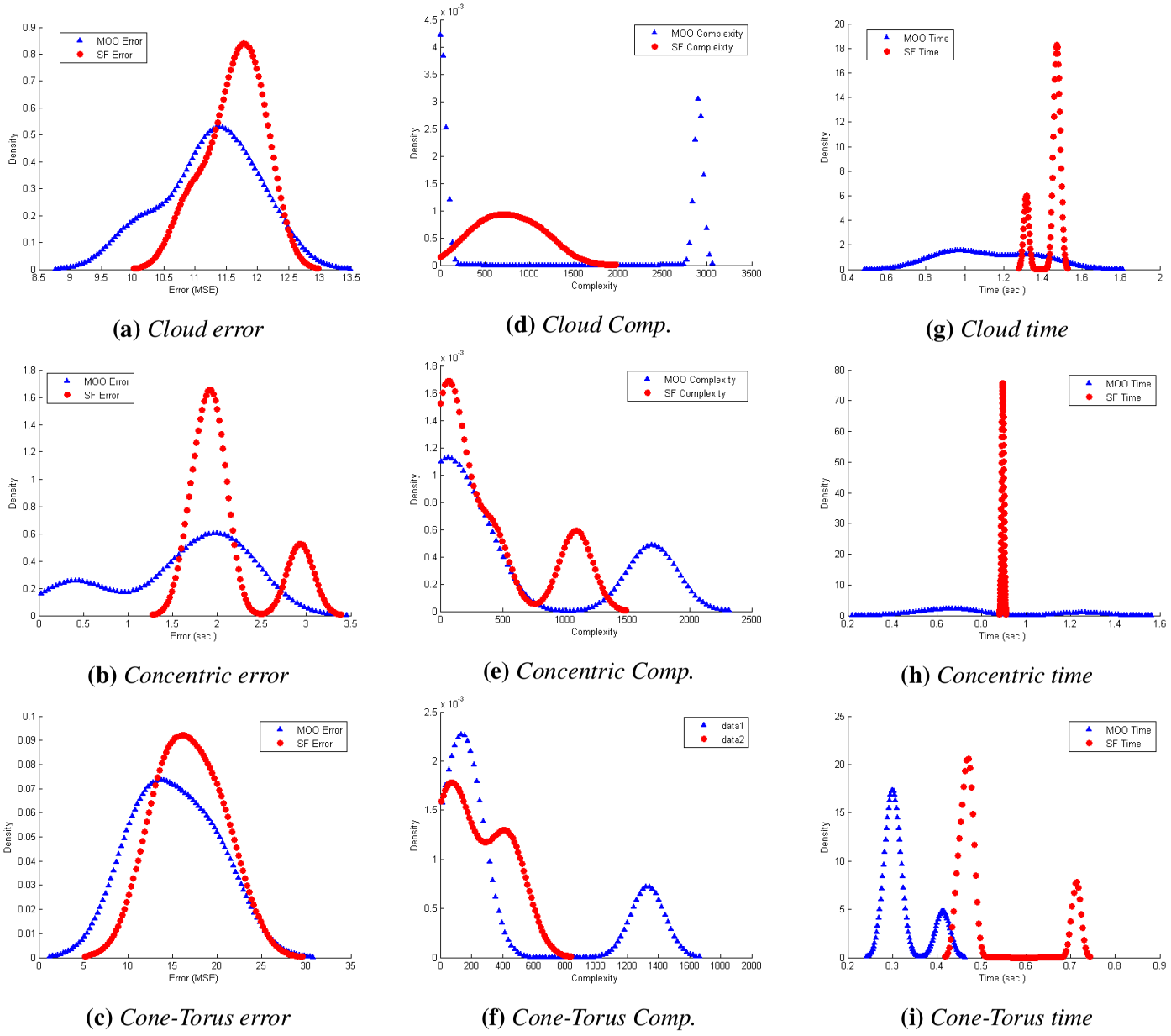
On the other hand, the execution time of the Pareto-based MOO approach is shorter than that of the scalarized MOO approach on four of the data sets (shown in Figures 3.4 (g), 3.4 (h), 3.4 (i) and 3.4 (x)). However, some of the non-dominated predictive models have longer execution time when applied to the Gaussians data sets as well as the shuttle data set. This is because Gaussians data sets have heavily overlapped distributions and are not linearly separable, while the shuttle data set is a large multi-class data set and the majority of its examples (80%) belong to one class. The trade-off between the accuracy, model complexity and the execution time which the Pareto-based MOO achieves; results in some of the non-dominated models having longer execution time but with a low error and/or complexity values. However, the time required by the NSGA-II to find the non-dominated models is long.

For a small population and a limited number of generations, a smaller number of non-dominated models can be found. For instance, in the previous experiment (discussed in Section 3.2) when the population size and the maximum number of generation were set to 10, only four to five models were found. The accuracy, model complexity and algorithmic complexity of these models were not as good as that of the models found in this new setting. As the population size and the maximum number of generations are increased from 10 to 100, the number of the non-dominated models was increased to 17 for the shuttle data set and 35 for all the remaining data sets. However, this comes at the cost of high algorithmic complexity. From this discussion it can be concluded that, choosing the population size and the number of generations defines the algorithmic complexity and the quality of the models found.

3.3.4 Limitations of the optimisation approaches

In general, the Pareto-based MOO approach requires all of the criteria to be measured independently, while this might not always be the case in the scalarized MOO approach. Measuring different criteria independently is not always feasible. For example, the model complexity of an SVM cannot be measured independently of its accuracy, as both criteria are optimised together during the training phase. However, these criteria can be measured for many prediction model families, for instance, decision trees can be used in the same framework where its model complexity can be measured as the depth of the tree, the accuracy and algorithmic complexity can be measured similarly.

One of the main drawbacks of the scalarized MOO is the need to define additional information, like providing the weights vector for the weighted sum method or ranking the criteria before applying ranking methods (Freitas (2004b)). Another drawback is that each criterion is measured using different units and they often have different scales, combining them in a single equation result in mixing different types of measurements. Furthermore, normalising the criteria defines a single scale, but it can result in losing the sense of how good or bad the criterion value is.

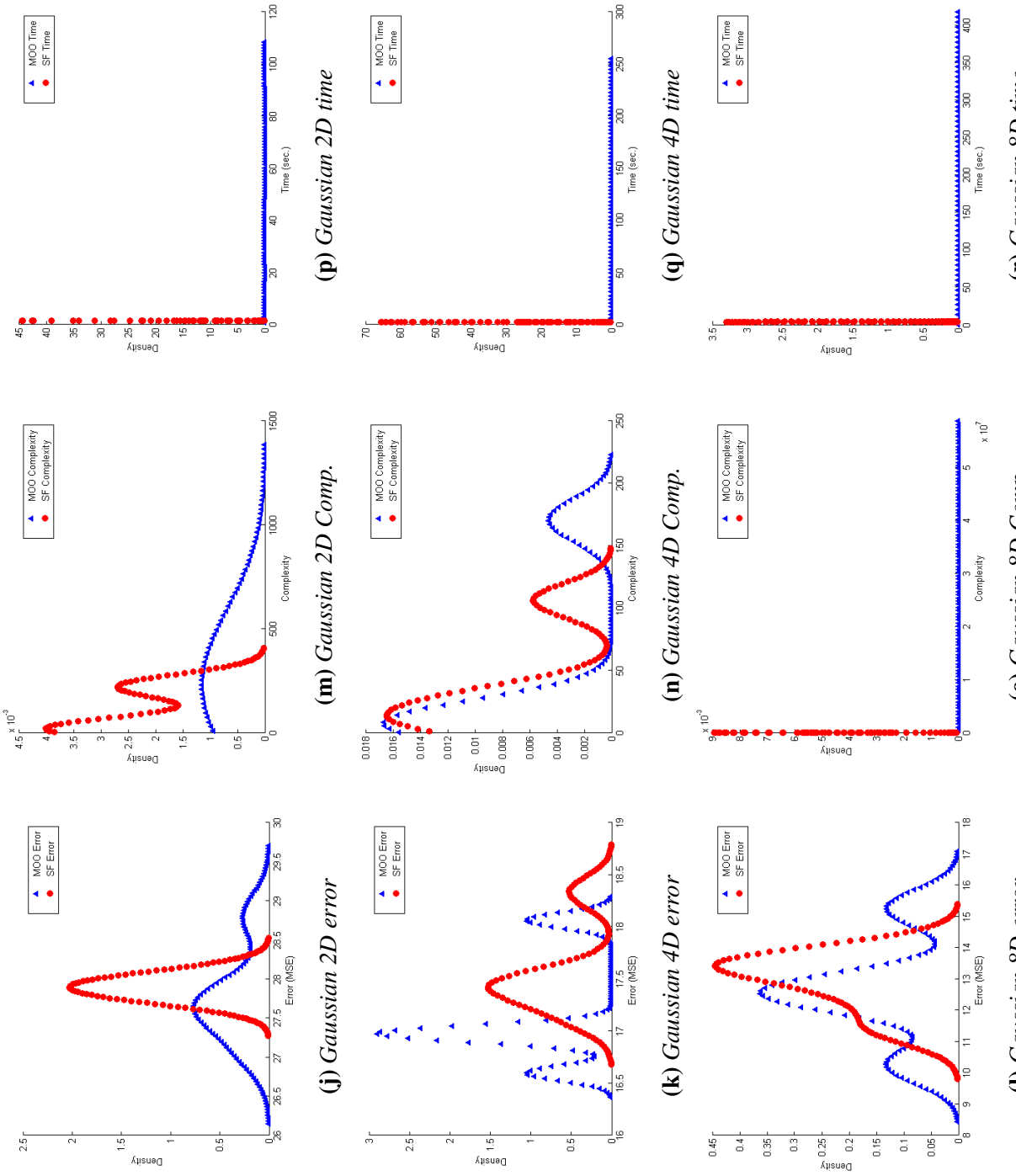


(a) *Cloud error* **(d)** *Cloud Comp.* **(g)** *Cloud time*

(b) *Concentric error* **(e)** *Concentric Comp.* **(h)** *Concentric time*

(c) *Cone-Torus error* **(f)** *Cone-Torus Comp.* **(i)** *Cone-Torus time*

Figure 3.3: *a) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 10 population size and a maximum number of generation of 10).*

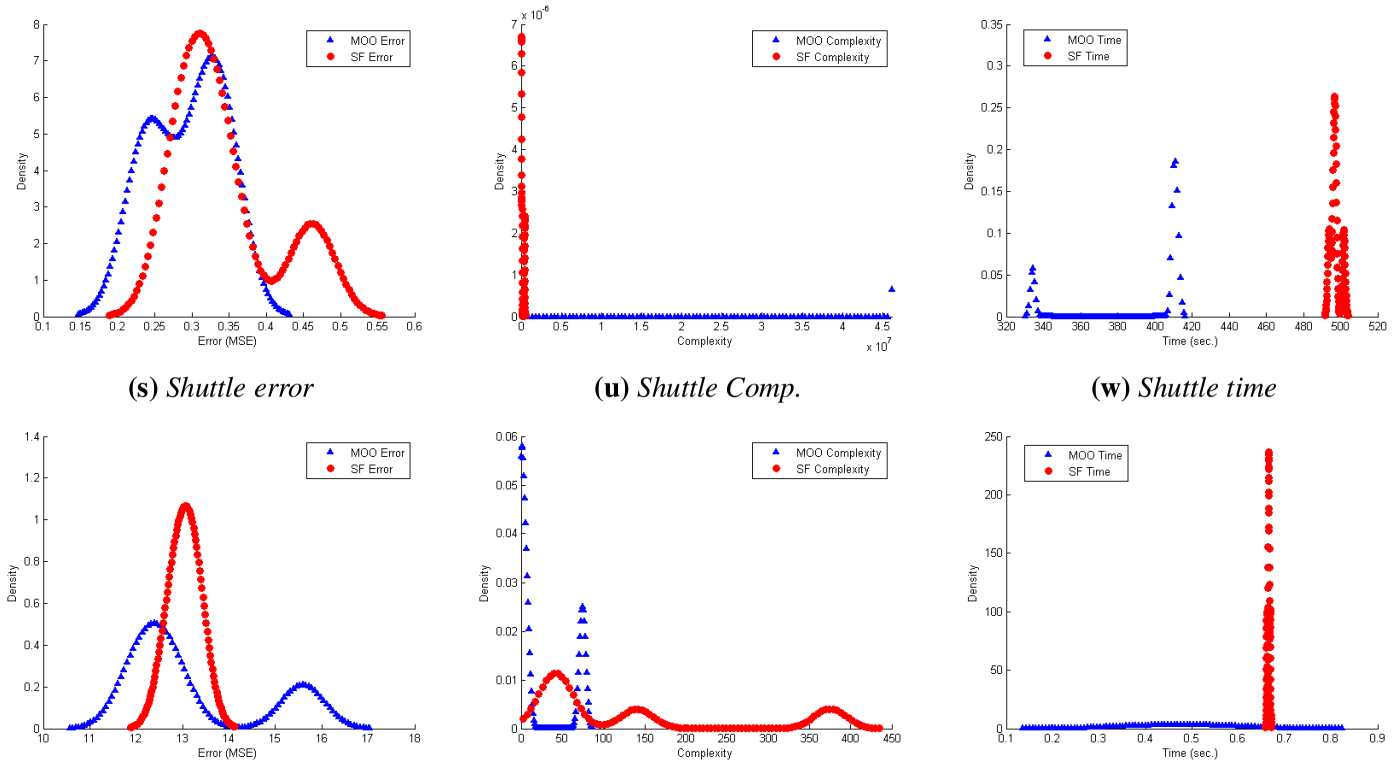


(j) *Gaussian 2D error* **(k)** *Gaussian 4D error* **(l)** *Gaussian 8D error*

(m) *Gaussian 2D Comp.* **(n)** *Gaussian 4D Comp.* **(o)** *Gaussian 8D Comp.*

(p) *Gaussian 2D time* **(q)** *Gaussian 4D time* **(r)** *Gaussian 8D time*

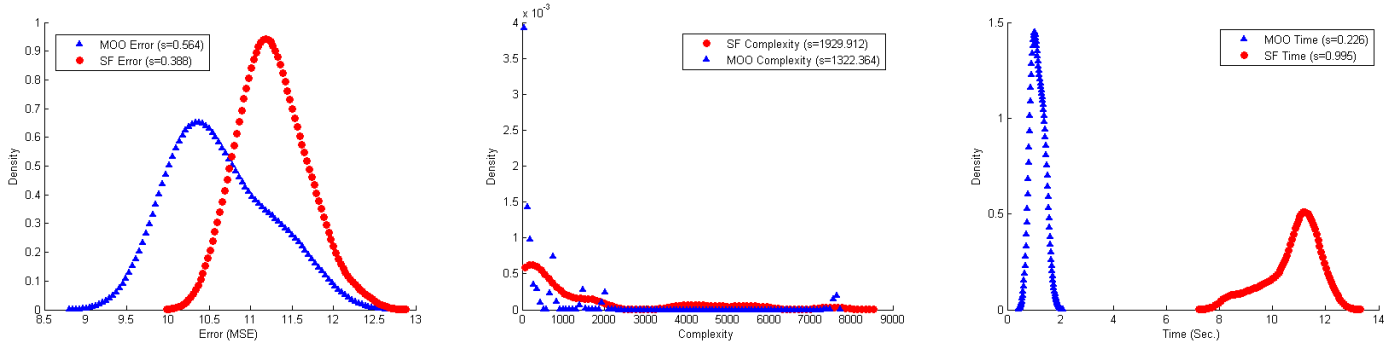
Figure 3.3: *b) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 10 population size and a maximum number of generation of 10).*



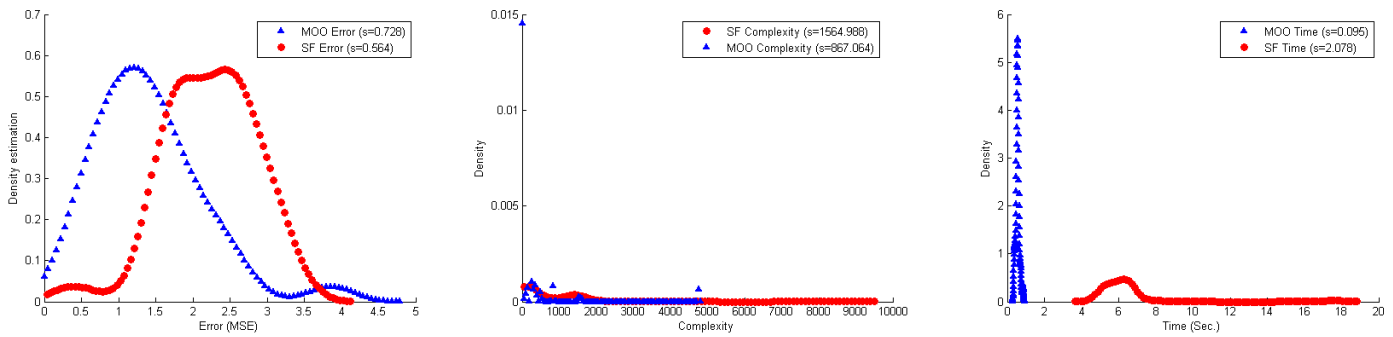
(s) *Shuttle error* **(u)** *Shuttle Comp.* **(w)** *Shuttle time*

(t) *Synthetic error* **(v)** *Synthetic Comp.* **(x)** *Synthetic time*

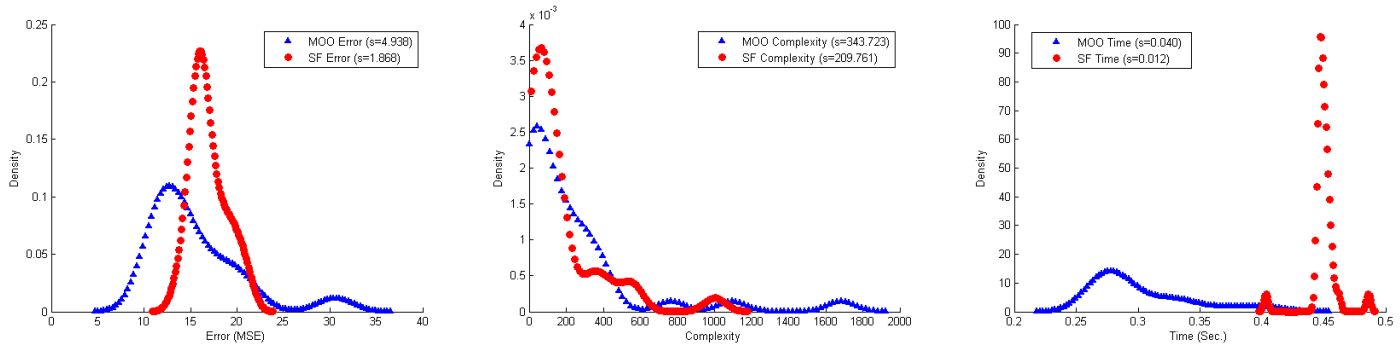
Figure 3.3: *c) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 10 population size and a maximum number of generation of 10).*



(a) *Cloud error* **(d)** *Cloud Comp.* **(g)** *Cloud time*

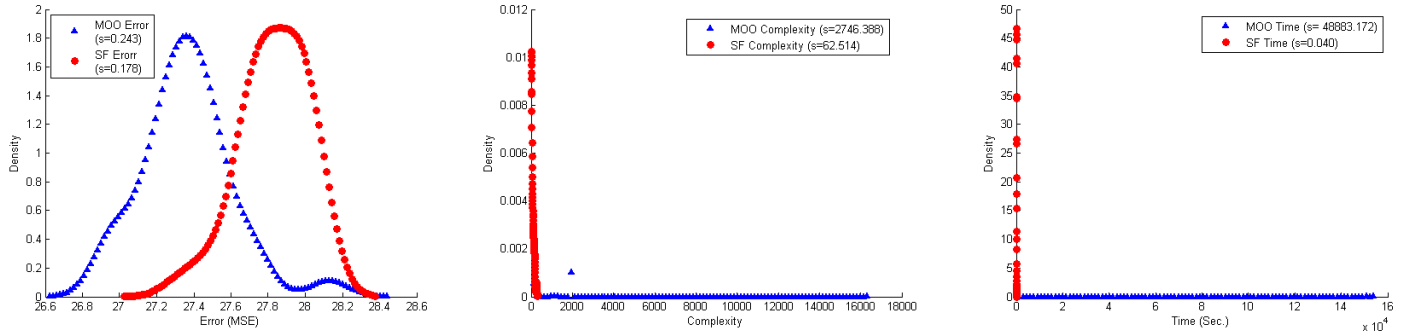


(b) *Concentric error* **(e)** *Concentric Comp.* **(h)** *Concentric time*

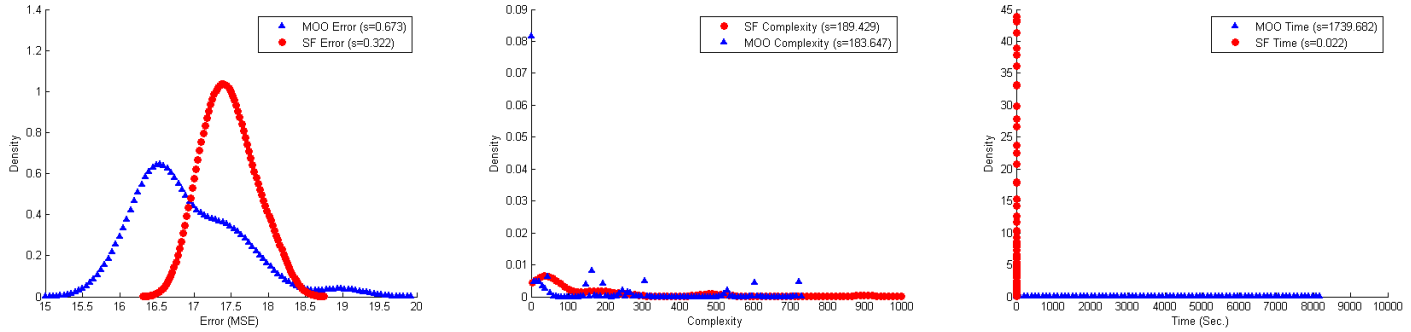


(c) *Cone-Torus error* **(f)** *Cone-Torus Comp.* **(i)** *Cone-Torus time*

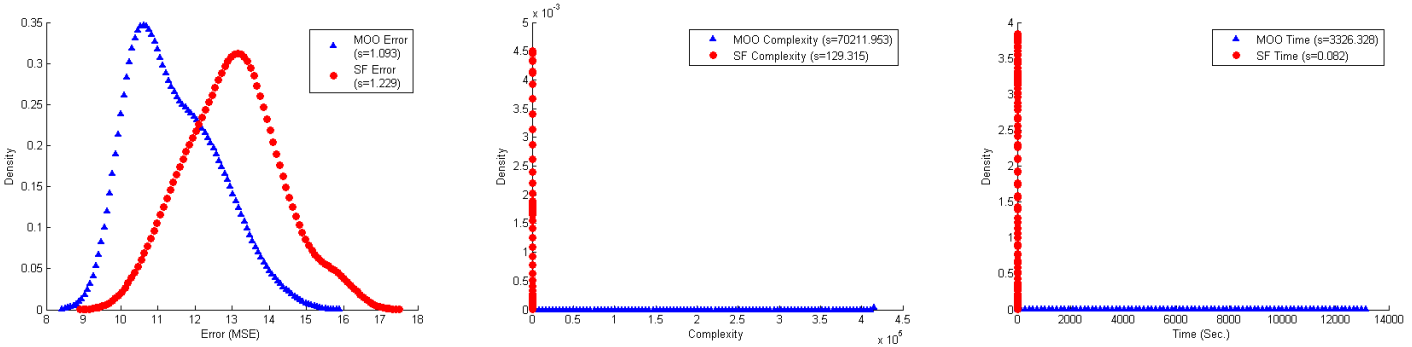
Figure 3.4: *a) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 100 population size and a maximum number of generation of 100).*



(j) *Gaussian 2D error* **(m)** *Gaussian 2D Comp.* **(p)** *Gaussian 2D time*

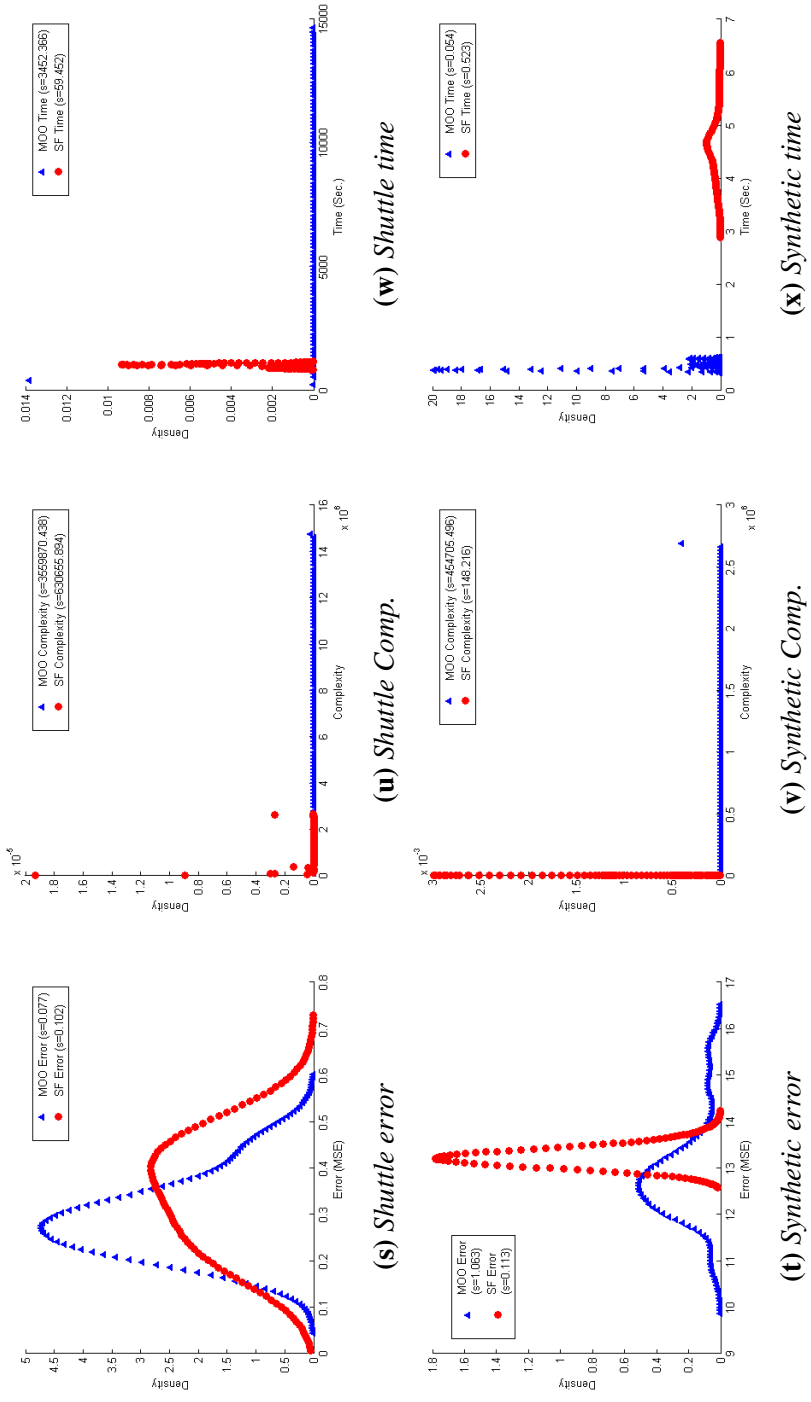


(k) *Gaussian 4D error* **(n)** *Gaussian 4D Comp.* **(q)** *Gaussian 4D time*



(l) *Gaussian 8D error* **(o)** *Gaussian 8D Comp.* **(r)** *Gaussian 8D time*

Figure 3.4: *b) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 100 population size and a maximum number of generation of 100).*



(s) *Shuttle error* **(u)** *Shuttle Comp.* **(w)** *Shuttle time*

(t) *Synthetic error* **(v)** *Synthetic Comp.* **(x)** *Synthetic time*

Figure 3.4: *c) Comparing the predictive model accuracy (MSE), model complexity and execution time of the models generated using the scalarized MOO approach and the Pareto-based MOO approach (with 100 population size and a maximum number of generation of 100).*

3.4 Summary

This Chapter introduces the design cycle of MCMLPS and investigates the generation of the base models for such systems using multiple criteria. The MCMLPS design cycle is presented in Section 3.2, it considers the evaluation of the base models using single as well as multiple criteria. Furthermore, this Chapter discusses the ability of defining universal measures for these criteria and the possible approaches for optimising them.
To explore the advantages and drawbacks of optimising the performance of the MCMLPS base predictors using multiple criteria, a comparative case study is presented in Section 3.3. In this case study both scalarized and Pareto-based MOO approaches are used to optimise the models using the accuracy, model and algorithmic complexities. Though in most cases the best models generated from the Pareto-based approach can have lower error than the models generated from the scalarized approach, nevertheless, the Pareto-based approach is hindered by its high algorithmic complexity. In general, using complex optimisation approaches and including more than one criterion in the generation of predictive models is a time consuming process. Thus in MCMLPS the performance of the base predictors is often evaluated using only the accuracy of the models prediction. While other characteristics of the system are either included indirectly in the design cycle or presented as constraints.
Moreover, the design cycle in this Chapter had examined the generation of local and global models. Chapter 4 will investigate the use of local models in building MCMLPS as well as the effect of encouraging diversity among the system base components.

Chapter 4

Diversity in Multi-component, Multi-layer Predictive System

4.1 Introduction

Population diversity had played an active role in the success of many methods. In natural selection, variation or diversity is one of the four main principles of the process. Natural selection aims to define the mechanism of evolution and it has inspired the development of many evolutionary algorithms which have been applied successfully to machine learning problems (Mitchell (1998)). Moreover, in ensemble learning diversity had been acknowledged as an important characteristic (Cunningham and Carney (2000), Lam (2000) and Krogh and Vedelsby (1995)). In literature, it has been shown that using an ensemble of predictors can often improve the generalization performance compared to that of a single predictor. The conditions for this improvement is for the base predictors to be diverse (their error correlation is reduced) and that they have a reasonable performance level (Jacobs (1995), Meir (1995), Opitz and Shavlik (1996a) and Tumer and Ghosh (1996)).
An ensemble with diverse models can have better performance due to the complementary behaviour of its components (Xue et al. (2006)). Furthermore, using ensembles can prevent the loss of information that results from choosing a single best model.
This chapter discusses diversity as a characteristic of MCMLPS, and investigates its effect on the accuracy of prediction. It starts by introducing the general structure of MCMLPS in Section 4.2. Then, different categorizations of diversity creation methods are explored in Section 4.3. Furthermore, Section 4.4 examines the methods used to gen-

erate diverse models. Next, locally weighted predictive systems are compared in Section 4.5. The design process of the MCMLPS presented in this Chapter is given in Section 4.6. This Section also discusses the use of two similarity metrics to generate the base components for the MCMLPS and examines the relation between the overall accuracy of the proposed system and the amount of disagreements among its base predictors. Finally, Section 4.7 summarises this Chapter.

4.2 Multi-Component, Multi-Layer Predictive Systems

The use of MCMLPSs have shown many theoretical and practical benefits compared to the use of a single best model. These include (Polikar (2006)): statistical benefits, as combining the output of several classifiers can often compensate for the possible unfortunate poor prediction of a single predictor, also, it is beneficial to use MCMLPS when the data is too large or too small. Furthermore, when the problem is too difficult to solve by a single predictor, MCMLPS can provide a divide and conquer strategy that a single predictor is incapable of achieving. Finally, when the data is generated from different sources (data fusion) a single predictor cannot represent the whole data accurately.
Once the base predictors of the MCMLPS are generated either a chosen set or all predictors are combined together. In general, combining methods can be classified according to their ability to train (trainable vs. untrainable combiners) and to the type of their output (class label vs. continuous output combiners) (Polikar (2006) and Ruta and Gabrys (2000)). One of the most widely used combining methods is the majority vote. It has been shown in (Ruta and Gabrys (2002)) that by suitable organising of the predictive system into multi-component, multi-layer structure the limits of the majority vote error for such a system can be significantly expanded in comparison to a traditional single layer ensemble. The theoretical findings from (Ruta and Gabrys (2002)) prompted the design of multistage selection fusion model discussed in (Ruta and Gabrys (2005)) and subsequent very successful extensions and applications of multi-component, multi-layer systems in time series forecasting (Ruta et al. (2011)) and generic predictive modelling (Kadlec and Gabrys (2009)) with examples of applications to airlines ticket demand prediction (Riedel and Gabrys (2009)), water pollution monitoring end prediction (Budka et al. (2010)) and adaptable soft sensors development in process industry (Kadlec and Gabrys (2009)).

The multi-layer, multi-component predictive system used in this study is shown in Figure 4.1, where w_{11},..., w_{nk}, are the weights of the first layer, n represent the number of the base ensembles and k represent the number of the models inside the base ensembles. On the other hand, w_1,..., w_n are the weights of the second layer for the n base ensembles. $M_1, ..., M_k$ are the base predictors of the first layer ensembles, $g_1, ..., g_n$ are the ensembles created from combining the base predictors and $\hat{Y}$ is the final prediction of the system.

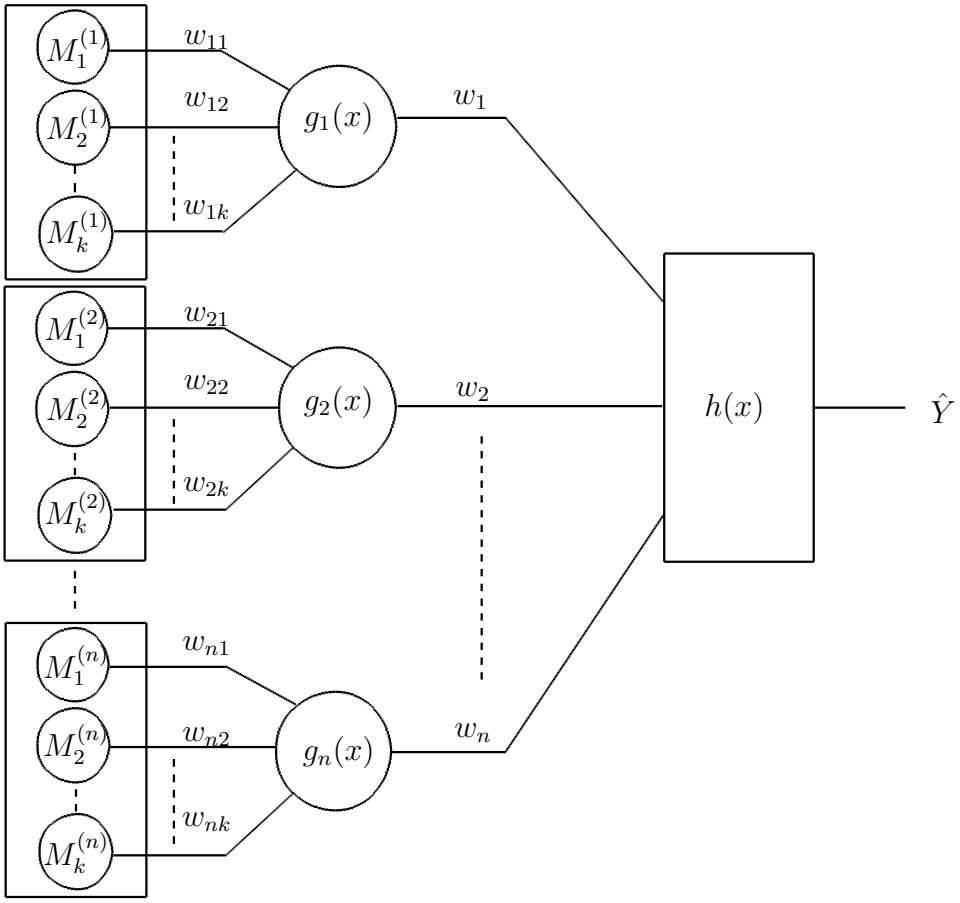


Figure 4.1: *General structure for the multi-component, multi-layer predictive system.*

Let X be the data set containing the training objects, C represent the number of classes, θ_c represent the actual class and M_k^n represent the output prediction of the model, where $M_k^n = 1$ for class θ_c and 0 otherwise and $c = 1, .., C$. The outputs of the base predictors M_k^n and the ensemble g_n are given as c-dimensional binary vectors where $[M_1^j, .., M_k^j]^T \in \{0, 1\}^c$ and $[g_1, .., g_j]^T \in [0, 1]^c$, j=1,...,n respectively. Equations 4.1 and 4.2 shows the mathematical representation for the ensembles generated from the first

layer:

$$g_j(x) = \Sigma_{i=1}^{k} w_{1j} M_i^{(j)}(x) \tag{4.1}$$

and let

$$d_{j,c}(x) = \begin{cases} 1 & if\ g_j(x)\ = \theta_c, \\ 0 & otherwise. \end{cases} \tag{4.2}$$

Then the second layer ensemble is as:

$$h(x) = \Sigma_{j=1}^{m} w_{2j} d_{j,c} \tag{4.3}$$

and the final prediction of the system is:

$$\hat{Y} = \arg\max_{c} h(x). \tag{4.4}$$

4.3 Diversity Methods Categorization

Several studies introduced different categorization approaches for diversity creation methods. In Brown et al. (2005) diversity creation methods for classification problems are classified as: explicit methods, like boosting where in each iteration a new classifier is built using different version of the training data, or implicit methods, like bagging, where each classifier is trained on a different randomly sampled set of the training data.
Another approach for categorizing diversity methods is to divide the search space of the predictors combination into coverage optimization and decision optimization (Ho (2001)). Where in coverage optimization the fusion method is fixed and the focus is on generating a set of mutually complementary predictors, while in the decision optimization the base predictors are fixed and the focus is on the fusion method.
In Kuncheva and Whitaker (2003), the diversity measures have been classified into pairwise and non-pairwise measures. In pairwise measures, the average distance between all classifiers in an ensemble is measured using a particular metric. On the other hand, non-pairwise measures use either entropy concept or the correlation between the classifiers and the average output.
Though the diversity among the ensemble classifiers is an important condition for the improvement of the ensemble generalisation ability, it has been shown in literature (Kuncheva and Whitaker (2003)) that using current diversity measures on their own does

not give an accurate prediction of the ensemble accuracy.
In the following subsection the main approaches used to generate diverse models are discussed.

4.4 Generating Diverse Models

There are a number of approaches that can be used to introduce diversity into ensemble base predictors. The following points summarise the main approaches used for this purpose:

1. Varying the initial condition: start each predictor with different randomly generated position in the search space. Though this method is widely used in the literature, it is seen as the least significant method for generating diverse predictors (Brown et al. (2005)). It shows no or only slight improvement in the generalisation error when applied.
2. Varying model architecture or model type: in this approach compatible learners are chosen to be combined in the ensemble. Examples of this approach are presented in (Islam et al. (2003) and Opitz and Shavlik (1996b)). In the case of incompatible learners (hybrid ensembles) where the ensemble consist of more than one type of predictive models, it is often the case that a single best model is chosen to provide prediction for each new instance. Examples of this approach are found in (Wang et al. (2000) and Langdon et al. (2002)).
3. Varying the training data: in this approach each predictor is trained on a subset of the training data or/and a subset of the features. This approach is more likely to generate diverse models than the previous two approaches (Xue et al. (2006)). The sets generated from this approach are either intersected sets (with overlapping instances) or disjoint sets (with non-overlapping instances). Learners trained on disjoint sets are more diverse; however, generating disjoint sets are often impractical in real world problems due to limited data.

Once the models have been created, a fusion method is used to combine them into a single ensemble. There are many fusion strategies such as: Majority vote, Borda count, threshold vote, heuristic decision rule, weighted average, fuzzy integral and fuzzy mode among others (Xue et al. (2006)).
This chapter investigate varying the training data to generate diverse ensembles, where

predictive models are trained on subsets of the data and/or subsets of the features. The following section discusses feature extraction and feature selection methods used to generate a representative subset of the data's features.

4.4.1 Feature extraction and feature selection

One way of generating diverse ensembles is to train the base predictors on subsets of the data or subsets of the features. In the case of training using subset of the data, the predictors do not see all of the available data which might be impractical in real case scenarios where limited data is available. On the other hand, in the second case each predictor is trained using a subset of the features. Reducing the number of features can be particularly helpful when the task at hand has a large number of features, many of which might be irrelevant to the task or redundant with respect to the other features (Brown et al. (2012)). In such cases using all of the available features to train a model can result in overfitting and can have a high computational cost. Reducing the number of features can be achieved by using feature extraction or feature selection methods (Xue et al. (2006)). Using feature extraction methods, such as Principle Component Analysis (PCA) or Independent Component Analysis (ICA), the dimensionality of the data is reduced by creating new features that represent the projections of multiple existing features. PCA mainly aim to find the features that contribute most to the variance (energy), and does not optimise for the class separability (Guo and Nixon (2009) and Bishop (1995)). Furthermore, as the information contained in the original feature set are projected into fewer principle components, there is a high risk of training the base predictors on the same or similar set of principle components, which eventually reduces the diversity among the predictors (Tumer and Oza (2003)).

On the other hand, using feature selection methods a different subset of features is chosen for the training of each base predictor. Features subset selection can be achieved through many approaches. Some popular choices of feature selection in classification problems are correlation (Tumer and Ghosh (1996)) and Mutual Information (MI) (Cover and Thomas (2012)). In (Tumer and Oza (2003)), an example that uses correlation based feature selection in classification problems is presented, this work focuses on the correlation between feature subset and the output classes (or a particular class) and aims to choose the set of features with the highest correlation. On the other hand, MI can be used to evaluate the dependencies between two features with respect to a certain class. This

approach has been applied in many pattern recognition problems, examples are the use of MI in selecting features for gait recognition problem (Guo and Nixon (2009)) as well as for medical signal selection (Deriche and Al-Ani (2001)).
The work presented in this chapter aims to divide the search space of the prediction problem (by selecting subsets of the features) into Local Regions (LRs) and train a set of local expert models on each LR. In order to generate the LRs two approaches are considered, the pairwise squared correlation and the conditional mutual information. In the first approach, similar features are grouped into one region, such that the predictive models trained on the resultant subsets specialize in a particular aspect of the prediction problem. The chosen features are the ones with the highest correlation (but they are not identical). The high correlation between the features is viewed as an indication for their similarity in defining a certain region of the search space. On the other hand, weakly correlated or independent features are assigned to different regions.
In the first approach a variation of Pearsons product-moment coefficient is used. Pearson's correlation method can only show linear-dependencies between the features. In this study a measure for higher order dependencies between the features is used, this measure is the correlation between the energy responses of the features. This measure was introduced in (Coates and Ng (2011)) and proved its efficiency in deep learning algorithms.
Meanwhile, in the second approach a number of LRs seeds are chosen using the conditional MI criterion. Then a modified version of this criterion is used to measure the similarity among the features and the LRs seeds, based on this criterion a subset of the features are assigned to each LR. The proposed criterion encourages the inner correlation between the features and the LRs seeds. The following sections consider and compare the methods used to generate local models. In addition, they provide a detailed description of the squared correlation approach and the conditional MI approach, how they are used to build MCMLPS and the results obtained when they are applied to a number of supervised classification problems.

4.5 Locally weighted predictive systems

In locally weighted predictive systems; the main motivation is to generate a set of base models that are both diverse and locally accurate. Such models will not have a uniformly consistent performance over the entire search space, but rather specialize in certain

regions of the prediction problem.

The MCMLPS proposed in this chapter is trained locally, where the locality is determined, as explained previously, using either the pairwise squared correlation between the features or the conditional MI among them. The base models of the proposed system are trained locally on disjoint sets of data and/or features. In the case when the correlation between the features is used as a metric to determine the locality, here, the local learners descriptor depend purely on information from the features of the training data which makes this method applicable in supervised as well as unsupervised learning. On the other hand, the conditional MI approach uses the output (class) of the data during the development of the system which makes this approach applicable only to supervised learning.

Once the LRs are generated, the proposed architecture is constructed as follow: first a sampling technique is applied to the data of each LR to generate a number of folds which represent the data within the region. Then a single model is trained on each fold of the LRs data. The LRs models are combined (using a weighted majority vote) into an ensemble. Finally the LRs ensembles are combined (using weighted majority vote) to provide the final prediction of the system. The weights of the ensemble's base predictors are calculated with respect to the method used for generating the data of the local regions (i.e. the squared correlation or the conditional mutual information of the features).

The proposed MCMLPS can be compared to the rotation forest (Rodriguez et al. (2006)) a well known algorithm which trains an ensemble of predictors on a subset of the features using a feature extraction method. The comparison is based on the method used to partition the data, the construction of the base predictors, how the descriptors (weights) of these base predictors are defined and how they are combined, as well as the results (training and testing accuracies and the disagreement between the base predictors). In the rotation forest the features of the data are split randomly into K disjoint subsets. Bootstrap algorithm is applied to 75% of the data in each subset, then PCA is applied to the bootstrapped data and the principle component coefficients are stored in the rotation array. Classifiers are built using (XR_l^a, Y) as the training sets, where X is the selected set of the data features, R_l^a is the rotation array and Y is the prediction outcomes. Prediction for any new data is delivered using the average combination of the classifiers. The following table compares the rotation forest to the MCMLPS constructed using squared correlation and conditional MI:

Table 4.1: *Comparing rotation forest, squared correlation and conditional MI ensemble methods .*

Operation	**Rotation Forest**	**Squared correlation LRs**	**Conditional MI LRs**
Data partitioning	Split features into K subsets	Split the data using the similarity of the features pairwise correlation into a pre-defined number of LRs	Split the data using features conditional MI into a pre-defined number of LRs
Learner descriptors	For each subset select a bootstrap sample of size 75% of the subset data and apply PCA to it. Store the principle components in a rotation matrix R_i^a	Use the squared correlation of the features	Use the conditional MI of the features
Base Predictors	Use (XR_l^a, Y) as the training sets for the base predictors	Train K base predictors on the LRs data	Train K base predictors on the LRs data
Combination Method	Sum of the classifiers predictions weighted by the rotation matrix $y_{final} = XR_l^a$	Majority vote weighted by the learner descriptor	Majority vote weighted by the learner descriptor

In the following sections the methodology of the squared correlation approach and the conditional MI are discussed.

4.6 Designing MCMLPS: Methodology

The procedure used to construct the MCMLPS presented in this chapter encompasses the following phases: a) data preparation and partitioning, b) model generation and combination. In this Section a detailed description of these phases is given.
In order to validate and examine the generalization ability of the proposed architecture, the Density Preserving Sampling (DPS) (Budka and Gabrys (2013)) is used to partition the data. DPS divide the data into subsets that are representative of the whole data set (Budka and Gabrys (2013)). In this work DPS is used to split the data into training and testing sets. The training data is assigned according to its features similarity to a set of LRs. The similarity is determined using one of two approaches: correlation based

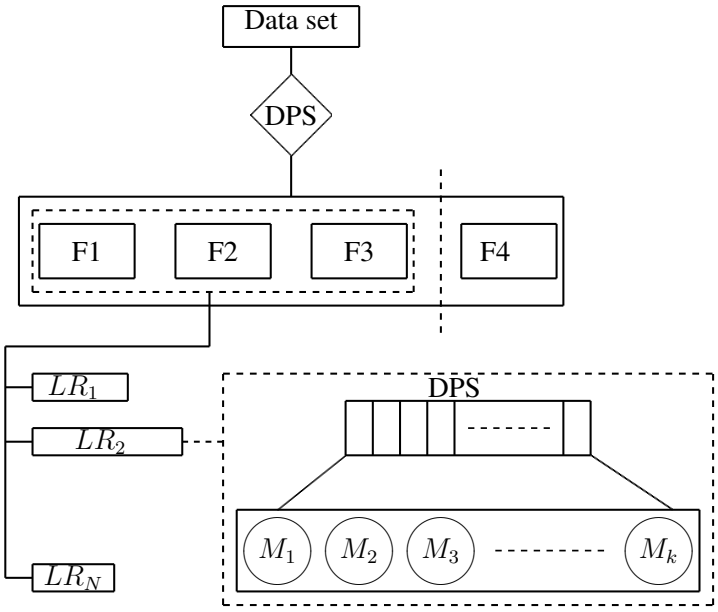


Figure 4.2: *Data preparation and model generation.*

approach and mutual information based approach (discussed in Subsection 4.1 and Subsection 4.2 respectively). Then DPS is used again to split the LRs data into K folds, where K models are trained on the data of the generated folds. The general design phases for the MCMLPS are discussed below:

- Data preparation and partitioning:
  After loading the data the following procedure is used to pre-process and partition the data. The data goes through three partitioning stages, first the whole data is split into training and testing sets, then the training set is allocated to the LRs and finally within the LRs the data is split into K subsets which are used to train the local models. Figure 4.2 shows the preparation and partitioning of the data, where, $F_1, ...F_4$ are the folds generated from the first DPS split, $LR_1, ...LR_N$ are the LRs and $M_1, ...M_k$ are the local models within the regions trained using data from the second DPS split.

  The points given below summarise the procedure used in this phase:

  - Apply DPS to split the data into 4 representative folds.
  - Use 3 out of 4 folds as the training data and the last fold as the testing data. Repeat for all four folds, so that each time a different fold is used for testing.
  - Find the similarity matrix for the training data using either the correlation

based approach or the mutual information based approach.

- Choose N rows from the similarity matrix to be the seeds for the LRs.
- Add the training data to the LRs according to the similarity of data features to the LRs seeds.
- Apply k fold DPS to the LRs data.

- Model generation, testing and combining:
  Once the data is assigned to the relevant LRs, the second DPS is applied to generate the K folds within the LRs and K models are trained on the LR folds. Furthermore, for all new instances N weights values are computed with respect to the N LRs. This phase can be summarized as follow:
  - Train a predictive model on each of the K LRs folds.
  - Compute the weights of the LRs votes using the similarity between the LRs seeds and the testing data. Given below is an illustrative example on how the weights of the LRs are calculated for a single instance. Where $f1...f7$ are the

Table 4.2: *Weights calculation for the illustrative example.*

f1	f2	f3	f4	f5	f6	f7	LRs weights
C1		C3		C5	C6		$C1 + C3 + C5 + C6$
	C2	C3			C6	C7	$C2 + C3 + C6 + C7$
C1	C2		C4			C7	$C1 + C2 + C4 + C7$

    features of the samples and $C1...C7$ are the similarity metric of the features. When a new sample arrive the values $C1...C7$ are computed and the summation of the selected features (with respect to the LRs) is used as weights for the corresponding LR's prediction.

  In the first layer, N ensembles are generated from combining the models of the N LRs. While, in the second layer a single ensemble that combines the first layer N ensembles is generated. The combining method used is a weighted majority vote with the similarity of the LRs features used as the weights in both layers.

4.6.1 Correlation based LRs

This approach aims to group similar features into the same LR. The similarity metric used is the pairwise square correlation between the features. This metric was introduced

in (Coates and Ng (2011)). The reason for using squared correlation is that if the data set consists of linearly uncorrelated features, then a higher measure of correlation between the features can be found by computing the energy correlation between two features at a time (the squared response).

Given a data set $X(i,z)$ where $X(i,:)$ represent the set of features that belong to a single sample, $i \in 1,...,I$ and $z \in 1,...,Z$. If $\mathbb{E}[x(:,z)] = 0$ and $\mathbb{E}[x(:,z)x(:,z)^T] = I$, also x_j represent the $x(:,j)$ feature and x_k represent the $x(:,k)$ feature then the following similarity measure between the squared responses of features can be defined:

$$S[x_j, x_k] = corr(x_j^2, x_k^2) = \mathbb{E}[x_j^2, x_k^2 - 1] / \sqrt{\mathbb{E}[x_j^4 - 1][x_k^4 - 1]} \tag{4.5}$$

The following points summarise the steps to generate the LRs using this metric:

- Whiten the input data set using Zero-phase Component Analysis (ZCA) whitening (Bell and Sejnowski (1997)).
- Compute the pairwise similarity between all the features using the following equation and store the results in the similarity matrix:

$$S_{j,k} \equiv S_X[x_j, x_k] \equiv \sum_i x(i,j)^2 x(i,k)^2 - 1 / \sqrt{\sum_i (x(i,j)^2 - 1) \sum_i (x(i,k)^2 - 1)} \tag{4.6}$$

- Select N rows, $j_1, ..., j_N$ of the similarity matrix S.
- Construct LRs containing the top M values of $S_{j,k}$
- Compute the pairwise squared correlation for the features of each training instance and compare the results with the seeds of the N LRs. Add the instance to the LR that has the highest similarity with respect to its feature values.
- Repeat for all N

Each one of the N rows serves as a seed for a single region. Once the seeds for the LRs are chosen, the pairwise squared correlation for the features of each new instance is computed and compared with the seeds. The instances are assigned to the LRs with the highest similarity. At the end of this stage an N disjoint sets (LRs) are constructed and using them a MCMLPS is built.

Note that applying the whitening procedure to the data removes the linear dependencies within the data. In such a case a measure of high order dependencies between two features can be obtained by looking at the correlation of their energy (squared responses). In

order to obtain the weights of the LRs ensemble prediction for the testing data, the pairwise squared correlation for each data instance is computed and similarity to the seeds of the LRs is measured. The summation of the energy for the corresponding features (to the LR) is used to weight the predictions of the LRs models (as was discussed in Table 4.2). In this approach initially both DPS and Cross Validation (CV) were used to split the LRs data. However, when comparing the accuracies obtained from the two methods it was found that the models trained on CV folds had large variation in their accuracies. On the other hand, the models that were trained on the folds generated from the DPS were more stable (i.e. they had lower variance in their error estimation). Due to this, DPS will be used in this work to partition the data.

4.6.1.1 Results

The proposed architecture described in the previous section is applied to the data sets shown in Table 4.3. The data sets used are taken from the UCI machine learning archive (Lichman (2013)). The base predictors used in the first set of experiments are CART Decision Trees (DTs) and in the second set of experiments are feedforward Neural Networks (NNs). The testing accuracies of the proposed architecture are compared to the accuracies of three benchmark algorithms, these are: Rotation Forest (RF), AdaBoost and Bagging.
The setting of the algorithms used in these experiments are given below. A predefined number of LRs and number of models within each LR is selected. The parameters are chosen for illustration purpose and so that the results obtained can be compared across all the data sets. Also, it highlights the advantages and drawbacks of predefining these parameters with respect to the data set size and dimensionality:

- Correlation based MCMLPS: 6 LR's are used each have 8 models (48 DT's in total) trained on disjoint subsets of the data. The number of features used in the LRs is determined through a separate optimization routine, where four different numbers of features are considered (with step size equal to the number of features$/4$) and the number of features that generate the maximum testing accuracy is chosen.
- RF: the number of classifiers are 6 and the number of disjoint features subspaces are 6.
- AdaBoost and Bagging: 48 DT were used as the weak learners for both algorithms.

Table 4.3: *Data sets details* .

Data sets	Features	Examples	Classes
Ionosphere	34	351	2
Pima	8	768	2
WBC	30	569	2
Heart	13	270	2
Sonar	60	208	2
Chess	36	3196	2
German credit card	24	1000	2
Spam base	57	4601	2
Gaussian 8D	8	5000	2
Vehicle	18	846	4
Waveform	40	5000	3

The following subsection discusses the internal accuracies of the proposed architecture and compare its performance with the benchmark algorithms. This is followed by two subsections that investigate the effect of changing the base predictors model type on the performance of both the RF and the MCMLPS architecture and the disagreement among their base predictors.

4.6.1.2 Internal Accuracies and Benchmark Comparison

The data sets used in these experiments are split using DPS into four folds, and each time a different fold is used for testing and the remaining three folds are used for training. The performance of the LRs models can vary over the four iterations and in most cases there is no single LR that dominates the others over the four iterations. Changing one fold of the training data can affect the performance of the LRs, where it can become better or worse and/or have a smaller or larger variation in its models accuracies. This is due to the new data instances being assigned differently to the LRs based on their pairwise similarity with the regions seeds. The term internal accuracy in this work refer to the variation in the accuracies of the LRs base predictors. An example of the internal accuracies of the LRs for the Gaussian 8 dimensional data set is shown in Figure 4.3.

It can be seen in Figure 4.3 that there is no single LR that has the best accuracy over the four iterations. Also the amount of variation in the accuracies of the individual LRs changes over the four folds, for example in iteration 2 the $6th$ LR has the smallest variation among its 8 models accuracies, however, for the same LR the amount of

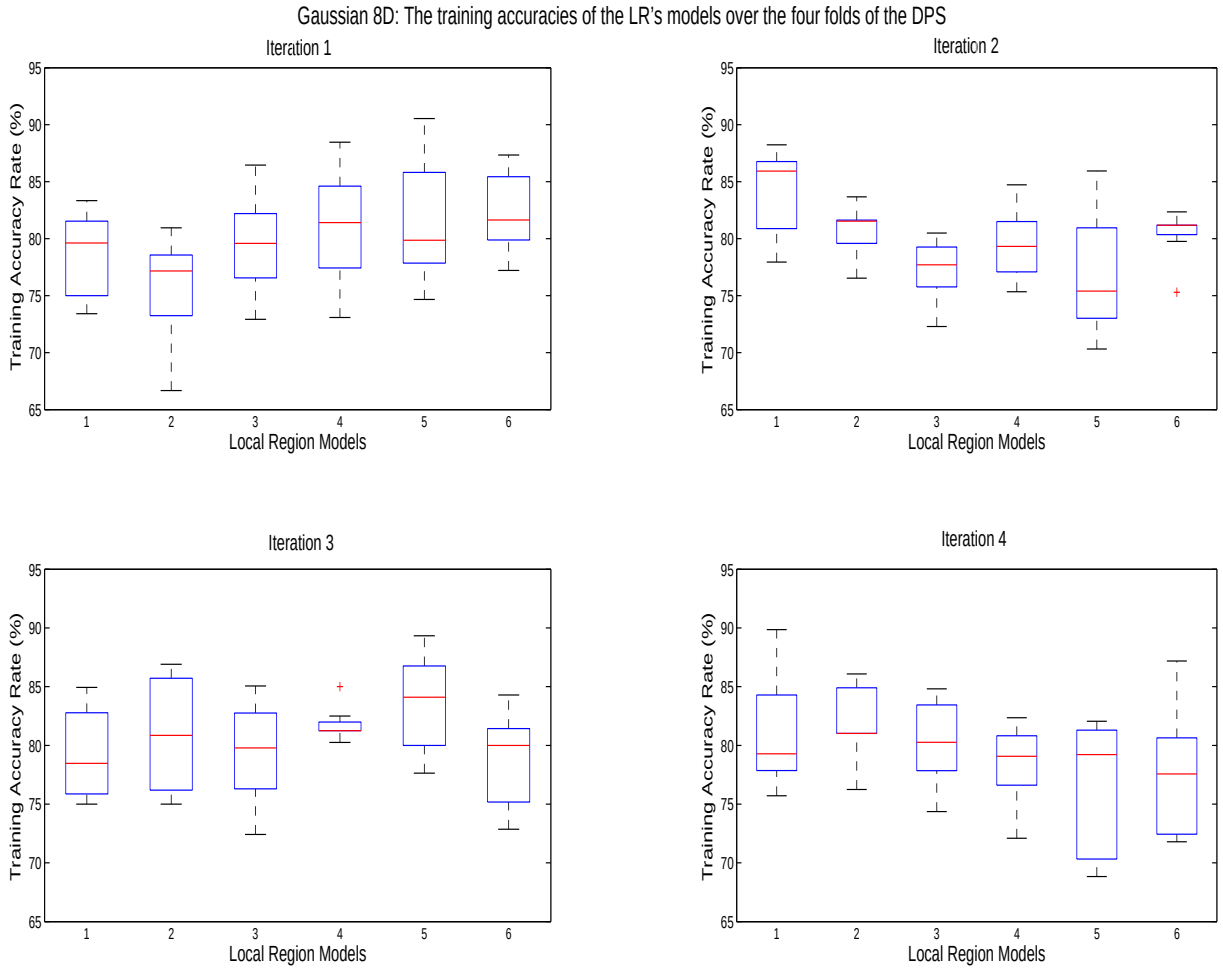


Figure 4.3: *Training accuracies of the local regions models for the correlation based MCMLPS when applied to Gaussian 8D data set.*

variation is higher in the other iterations. The same behaviour can be seen in the internal accuracies of the other data sets.

In general, large data sets lead to a more stable performance of the LRs and reduce the accuracy variation within and across the LRs. However, how big a data set should be, depends on the dimensionality of the data and its distribution in the search space. A small data set with high dimensionality like the ionosphere data set can have a wide range of variation in the LRs accuracies. This is due to the small number of instances that are assigned to the LRs, which can lead to a model being validated on few or a single data instance(s). Classifying this instance(s) correctly or incorrectly results in 0% or 100% accuracy. The solution to this problem is to lower the number of LRs and/or the

number of their internal models. In order to be able to compare the accuracies over all of the data sets used in these experiments, the same number of LRs and models are used.

Benchmark Comparison

The overall testing accuracies of the MCMLPS architecture are evaluated and compared with the three benchmark algorithms described above. Table 4.4 shows the average testing accuracy for the learning algorithms over the four DPS folds. Different types of pre-processing can be applied to the data sets before the benchmark ensemble methods are used. As has been discussed previously, in the correlation based MCMLPS the data sets are pre-processed using ZCA whitening procedure. This pre-processing method removes the linear dependencies among the data's features. On the other hand, in the RF, PCA is applied to obtain the values of the rotation matrix. In this set of experiments both bagging and AdaBoost are applied directly to the training data (no method for pre-processing the data is used). Applying ZCA to the data while using Bagging and AdaBoost methods resulted in a general decrease in their accuracies. This case was investigated in (Al-Jubouri and Gabrys (2016)) where we had applied the same type of pre-processing method to the proposed architecture as well as to five benchmark algorithms that were used (including both bagging and AdaBoost). Moreover, removing the whitening procedure (keeping the linear dependencies) in the MCMLPS proposed in this chapter can lead to unbalanced split of the data. In this case a number of the LRs can end up having very few samples to train the base predictors, which can result in overfitting and consequently decreases the overall accuracy of the system. In this experiment an additional parameter that represents the density of the data inside each LR is added to the weighing of the LRs prediction.

Some of the results shown in this Section for the RF method differ from the results obtained in (Rodriguez et al. (2006)) where the RF was first introduced and compared to both Bagging and Boosting methods. This is due to the use of different type and different number of base predictors. The aim of using different setting from the original paper is to have similar setting for all the benchmark ensemble methods which enable the comparison of their results.

As can be seen from Table 4.4, the proposed system does not have the best accuracies compared to the benchmark algorithms. Nevertheless, for certain data sets the obtained performance of the system is comparable to that of the other benchmark algorithms, while that is not the case for other data sets. Generally, with the current setting of

Table 4.4: *Comparing the test accuracies of the four ensemble methods when CART DTs are used as their base predictors.*

Data sets	correlation based MCMLPS	RF	Bagging	AdaBoost
Gaussian 8D	88.16	80.70	88.78	87.08
German credit cards	70.00	65.30	77.30	75.90
Ionosphere	77.19	92.61	93.44	93.16
Spambase	85.20	85.50	95.37	93.20
Pima Indians Diabetes	76.62	73.30	77.60	77.08
WBC	86.29	91.56	95.61	95.25
Heart	76.65	77.06	85.18	83.34
Sonar	63.46	74.04	87.02	83.17
Chess	93.74	70.46	98.99	94.84
Vehicle	67.61	61.37	77.07	51.07
Waveform	65.68	91.46	85.74	80.78

the proposed architecture and the current type of base predictors, the performance of the systems is highly influenced by the size of the data sets. The data in the proposed MCMLPS is split multiple times to generate the LRs and to train the base predictors within the LRs. Thus having a small number of samples can lead to base predictors overfitting the training data and result in decreasing the overall accuracy of the system compared to other benchmark methods. The correlation based MCMLPS also have the lowest accuracy on the waveform data set. Though the size of this data set is large, it is a multi-class problem with high dimensionality and when the type of the base predictor is changed from decision trees to NN the accuracy of this data set increased.

4.6.1.3 Changing the type of the base predictors

In this subsection the type of the base predictors is changed from CART DTs to feed forward NNs for both the MCMLPS architecture and the RF method. The accuracies of both methods using the new base predictor is shown in Table 4.5.

In general, changing the base predictors to NNs improved the overall testing accuracies for both methods. However, the improvement in the testing accuracy of the RF method is higher than that of the correlation based MCMLPS architecture. Moreover, the accuracy of the multi-class problem has improved much more than that of the binary classification problem (as can be seen in the accuracies of the waveform and the vehicle data sets).

Table 4.5: *Comparing the test accuracies of correlation based MCMLPS and RF when feedforward NNs are used as their base predictors.*

Data sets	Correlation based MCMLPS	RF
Gaussian 8D	88.45	88.40
German credit cards	70.00	70.00
Ionosphere	74.25	93.15
Spambase	90.55	85.75
Pima Indians diabetes	76.30	76.80
WBC	91.55	95.61
Heart	77.02	81.11
Sonar	62.50	79.81
Chess	96.75	73.06
Vehicle	78.50	81.75
Waveform	85.05	92.65

The test accuracies of the proposed architecture have improved on all but the ionosphere, pima Indian diabetes and sonar dataset, where they slightly deteriorated by 2.94%, 0.32% and 0.96% respectively. The performance remains the same on the German credit card data set.

4.6.1.4 Disagreements among the base predictors

In the proposed architecture, when CART DTs are used as the base predictors, there are varied levels of disagreements within the LRs models and even a higher level of disagreement across the LRs. Meanwhile, when feed-forward NNs are used as the base predictors, similar models are generated in the individual LRs, yet there is still a high level of disagreement across the LRs. The total disagreement values are found by measuring the disagreement between the final prediction of the system and the prediction of the individual LRs ensembles.

On the other hand, the total disagreement among the classifiers of the RF method is much lower than the proposed architecture. Figure 4.4 shows the total disagreement among the LRs of the MCMLPS and the classifiers of the RF when both NNs and DTs are used as the base predictors. In general, using NNs as the base predictors for the RF method results in higher disagreements among the classifiers than when CART DTs are used. One exception is the German credit card data set, where the disagreements within the RF classifiers are zero. That is because; all the classifiers tend to vote for the majority

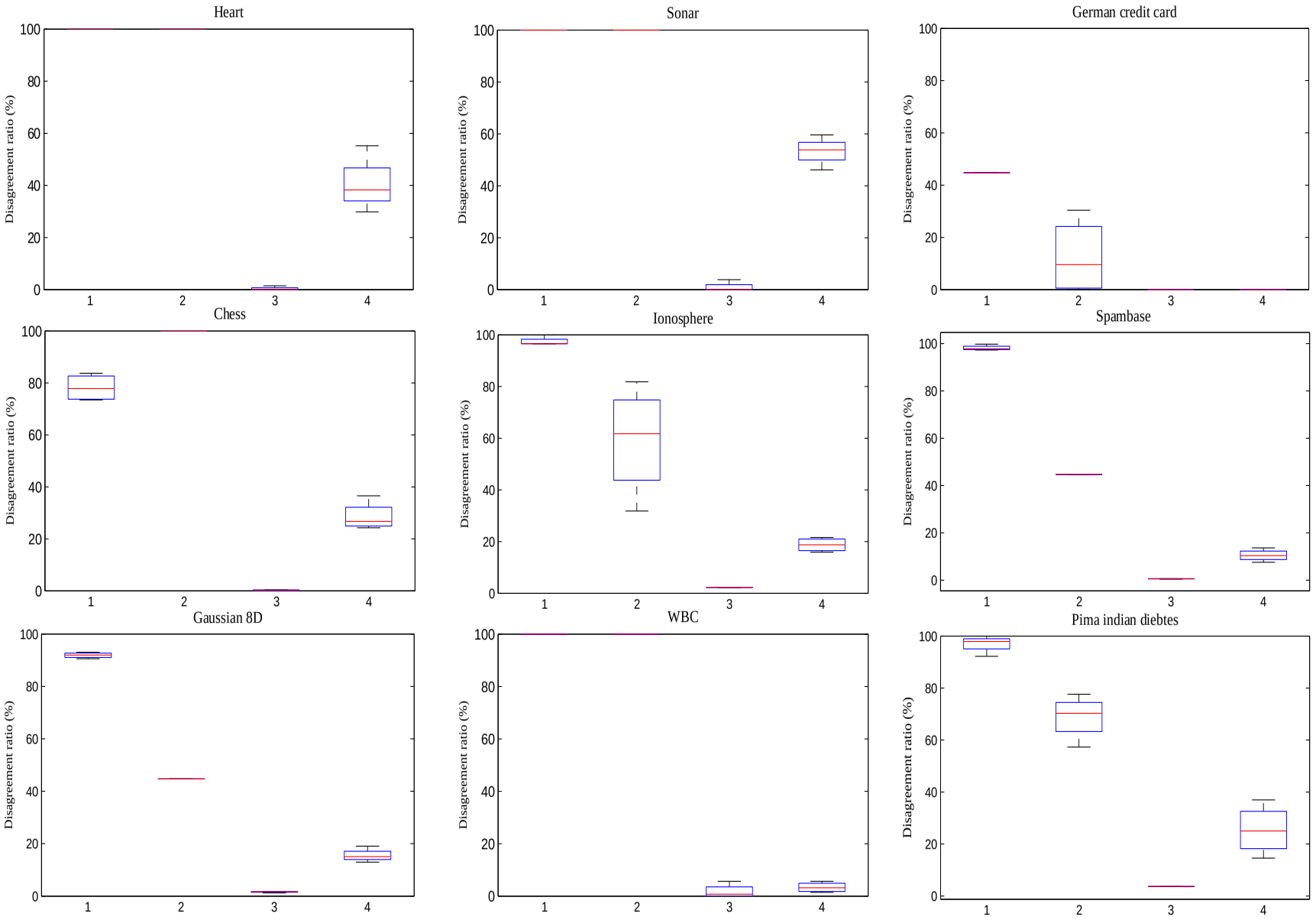


Figure 4.4: *comparing the disagreements among the classifier/LR of both RF and MCMLPS, the base predictors used are: NN and CART DTs., where* $1 =$*LR DT,* $2 =$ *LR NN,* $3 =$ *RF DT and* $4 =$ *RF NN*

class for this data set. On the other hand, in the MCMLPS the overall disagreement generated from using the DTs as the base predictors is generally higher than when NNs are used.

Though both algorithms have different levels of disagreements, yet their performance is comparable across most of the data sets used in these experiments. The high level of disagreement of the proposed architecture can be beneficial when applied on noisy data set (this will be explored in the next chapter where the robustness of the MCMLPS is discussed).

In the ionosphere data set changing one fold (a third) of the training data greatly affects the level of disagreement among the LRs when NNs are used. The difference between the iteration with the highest disagreement and the iteration with the lowest disagreement is equal to 50%. This again shows that, the size of the data set has high impact on the accuracies of the LRs. In the current setting of the MCMLPS, when small data sets are used, the LRs models do not have enough data to train on and that result in the high variations among the LRs models. In these experiments the parameters of the proposed MCMLPS were predefined in order to be able to compare the results with a number of benchmark algorithms, however, these parameters (such as the number of the LRs and the number of models within the LRs) can and should be optimized to suit the classification problem.

4.6.2 Conditional mutual information based LRs

The conditional mutual information based LRs aims to split the feature space into a number of subsets based on the Conditional Mutual Information (CMI) of the features. This approach and its main findings has been published in Al-Jubouri and Gabrys (2017). The features with the highest CMI values are chosen to be the seeds for the LRs. The CMI is measured using the following equation (Brown et al. (2012)):

$$J_{cmi}(X_k) = I(X_k;Y) - I(X_k;S) + I(X_k,S|Y) \tag{4.7}$$

where X_k is a single feature, Y is the output and S are the remaining features (all the features apart from X_k). $I(X_k;Y)$ is the mutual information between the feature X_k and the class Y, $I(X_k;S)$ is the redundancy of feature X_k with respect to the remaining

features and $I(X_k, S|Y)$ is the conditional redundancy (the class dependency of X_k with the existing feature set S). According to Brown et al. (2012) the equation given above shows that including correlated features can be useful, if the correlation of the features with the class is higher than their inner correlation. The benefits of including correlated features have been explored before by (Guyon et al. (2008)), where it has been observed that correlation does not imply redundancy.

Once the CMI values of the features are computed using equation 4.7, the highest N features are selected to be the seeds for the LRs. In order to add new features to the LRs, the similarity of the features to the LRs seeds need to be calculated. Equation 4.8 is used to determine the similarity between the features and the LRs seeds.

$$J_{cmi+}(X_k) = I(X_k; Y) + I(X_k; J_{cmi}(X_k)) + I(X_k, J_{cmi}(X_k)|Y) \tag{4.8}$$

In this equation the pairwise mutual information of the features with the LR seeds is calculated and the features that have the highest CMI with respect to the seeds are added to the LRs. By adding rather than subtracting the redundancy term $I(X_k; J_{cmi}(X_k))$ this approach aim to group together similar features in the LRs. Each LR is assigned with a subset of the features, where all the features are ranked according to their mutual information with the seed of the LR and only the highest ranking features are assigned to the LR. The ratio of the features assigned to the LRs is α, where $0 > \alpha > 1$.

At the end of this stage N subsets of features are assigned to the LRs. The following subsection describes the methodology of using this approach to build an MCMLPS. In order to build the MCMLPS the same methodology used in the previous approach is followed. Initially the data is split using the method presented in Figure 4.2. DPS is also used to split the data into training and testing. However, when CV is used instead of DPS, it showed less variation in the accuracy of the LRs models than it has shown in the previous approach. This case is investigated in Section 5.4.2 and its results are compared to the DPS results.

The following points summarise the steps used to split the training data into N LRs:

- Calculate the conditional mutual information among the training data features using equation 4.7.
- Choose the highest scoring N features to be the seeds of the LRs.
- For the remaining features, use equation 4.8 to rank the features according to their similarity to the LRs seeds.

- Based on the features mutual information with the seeds, assign α of the total number of features to the LRs.

The prediction of the LRs models are combined using a weighted majority vote in the first layer, then the prediction of first layer ensembles are combined also using weighted majority vote to provide the final prediction of the system. In both layers the mutual information of the LRs features is used as a weighting vector. The weight for the LRs models is calculated using the summation of the mutual information values of the LR features. The weights are computed similarly to the weights in the previous approach (see Table 4.2) except that instead of the squared correlation the mutual information values computed using equation 4.8 is used. This value is computed for the features of the LRs with respect to the seed.

4.6.2.1 Results

The MI based MCMLPS described in the previous section is applied to the data sets shown in Table 4.3. The performance of this system is compared to the correlation based MCMLPS, RF, bagging and AdaBoost. The setting for these benchmark algorithms is the same as in the previous study. In order to be able to compare the results obtained from this system with the correlation based MCMLPS, both the number of the LRs and the number of models inside the LRs are set to the same values used in the previous study (6 LRs with 8 models inside each one of the LRs). Furthermore the α value (the ratio of the features assigned to the LRs) is set to 30%. The base predictors used are CART DTs and feedforward NNs.
The following subsections discuss the internal accuracies of the LRs base predictors and compare the overall system performance with the benchmark algorithms. This is followed by a subsection that investigates the level of disagreement among the LRs prediction of the MCMLPS.

4.6.2.2 Internal accuracy and benchmark comparison

Similar to the correlation based MCMLPS, in this section the internal accuracies of the LRs base predictors are measured and compared across the four DPS folds. In general the variation in the LRs base predictors accuracy is less than previous approach. An example of the LRs base predictors internal accuracies for the Gaussian 8 dimensional data set is

shown in Figure 4.5.

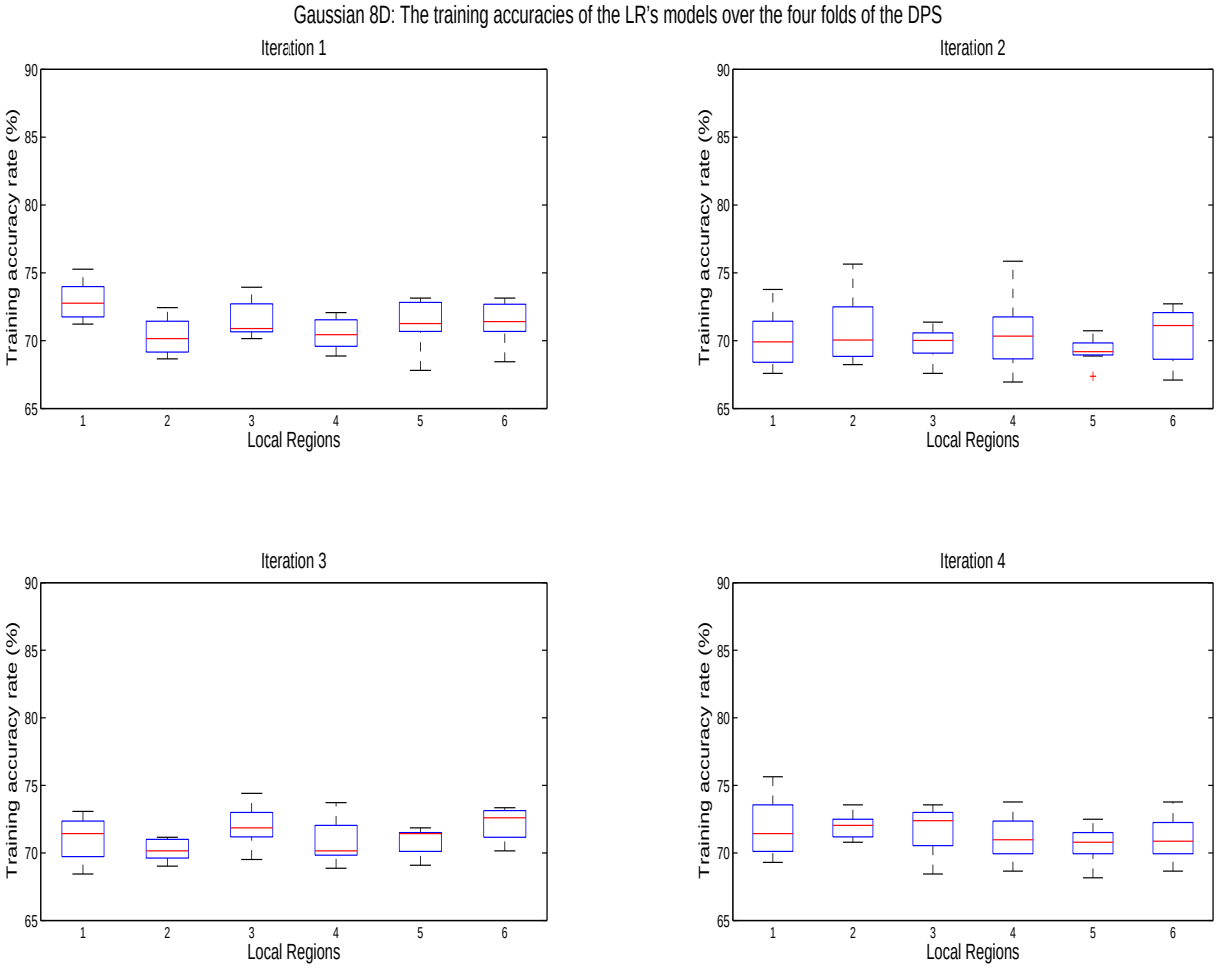


Figure 4.5: *Training accuracies of the local regions models for the MI based MCMLPS when applied to Gaussian 8D data set.*

Comparing Figure 4.5 and Figure 4.3 shows that the variance in the internal accuracies of the second architecture is much lower than that of the first architecture and once again there are no LR that outperform the rest on all of the four folds. In the MI approach even small data sets like the Ionosphere data set, has a lower variation in its internal accuracies than the previous approach. A possible explanation for this is that the LRs in this case are trained on subset of the features for all the data set rather than being trained on disjoint subsets of the data. Though the base predictors are trained on 30% of the total number of features, yet they can see all of the training data.

Table 4.6: *Comparing the test accuracies of the five ensemble methods when CART DTs are used as their base predictors.*

Data sets	MI based MCMLPS	correlation based MCMLPS	RF	Bagging	AdaBoost
Gaussian 8D	86.94	88.16	80.70	88.78	87.08
German	74.60	70.00	65.30	77.30	75.90
Ionosphere	92.30	77.19	92.61	93.44	93.16
Spam base	93.81	85.20	85.50	95.37	93.20
Pima	75.78	76.62	73.30	77.60	77.08
WBC	95.61	86.29	91.56	95.61	95.25
Heart	78.89	76.65	77.06	85.18	83.34
Sonar	84.62	63.46	74.04	87.02	83.17
Chess	98.78	93.74	70.46	98.99	94.84
Vehicle	74.35	67.61	61.37	77.07	51.07
Waveform	81.80	65.68	91.46	85.74	80.78

Benchmark comparison

The overall testing accuracy of the MI based MCMLPS averaged over the four DPS iterations are shown in Table 4.6. The results are compared to the correlation based MCMLPS as well as RF, Bagging and AdaBoost algorithms. It can be seen that, this approach for generating the LRs has generally improved the testing accuracy obtained from the correlation based MCMLPS. Compared to Bagging which has the highest accuracy for all the data sets (except for the Waveform data set) the difference between the test accuracy of this approach and the Bagging ranged between (0) for WBC data set and (6.29) for the heart data set.

Table 4.7 shows the test accuracy of the MI based MCMLPS compared to the correlation base MCMLPS and the RF, when the type of the base predictors are changed from CART DTs to feedforward NNs. With the RF the testing accuracy of this algorithm increases on every single data set when the feedforward NNs are used as the base predictors. Unlike the RF, the MI based MCMLPS showed mixed responses, where the accuracy increased for only (5 out of 11) data sets.

4.6.2.3 Disagreements among the base predictors

The disagreement among the LRs votes and the final prediction when CART DTs as well as feedforward NNs are used as the base predictors for the MI based MCMLPs are shown in Figures 4.6 and 4.7 respectively. It can be seen that the amount of disagreement

Table 4.7: *Comparing the test accuracies of MI based MCMLPS, correlation based MCMLPS and RF when feedforward NNs are used as their base predictors.*

Data sets	MI based MCMLPS	correlation based MCMLPS	RF
Gaussian 8D	84.22	88.45	88.4
German credit cards	77.50	70.00	70.00
Ionosphere	90.03	74.25	93.15
Spambase	90.44	90.55	85.75
Pima Indians Diabetes	76.04	76.30	76.80
WBC	94.90	91.55	95.61
Heart	82.61	77.02	81.12
Sonar	82.21	62.50	79.81
Chess	94.65	96.75	73.06
Vehicle	79.67	78.50	81.75
Waveform	85.36	85.05	92.65

when DTs are used is higher than when NNs are used as the base predictors for the LRs. However, in both cases, it is higher than the disagreement among the RF classifiers. As has been mentioned before, the robustness of such highly diverse system in noisy environment will be investigated in the following Chapter.

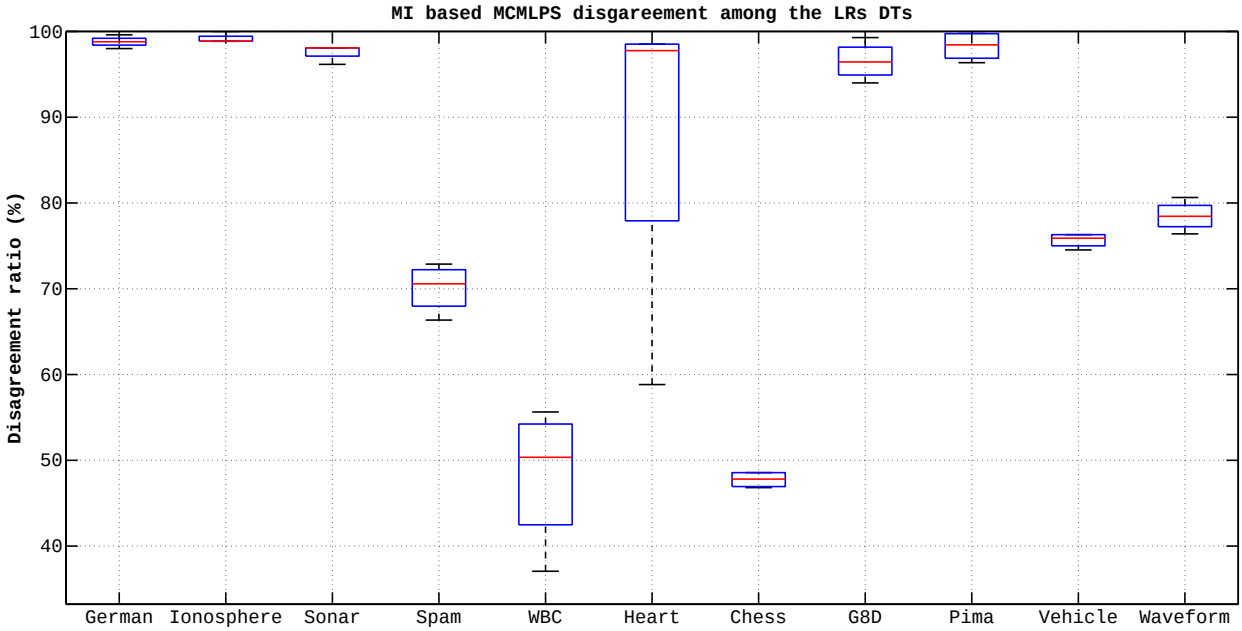


Figure 4.6: *Comparing the disagreements among the LRs of MI based MCMLPS when CART DTs are used as the base predictors.*

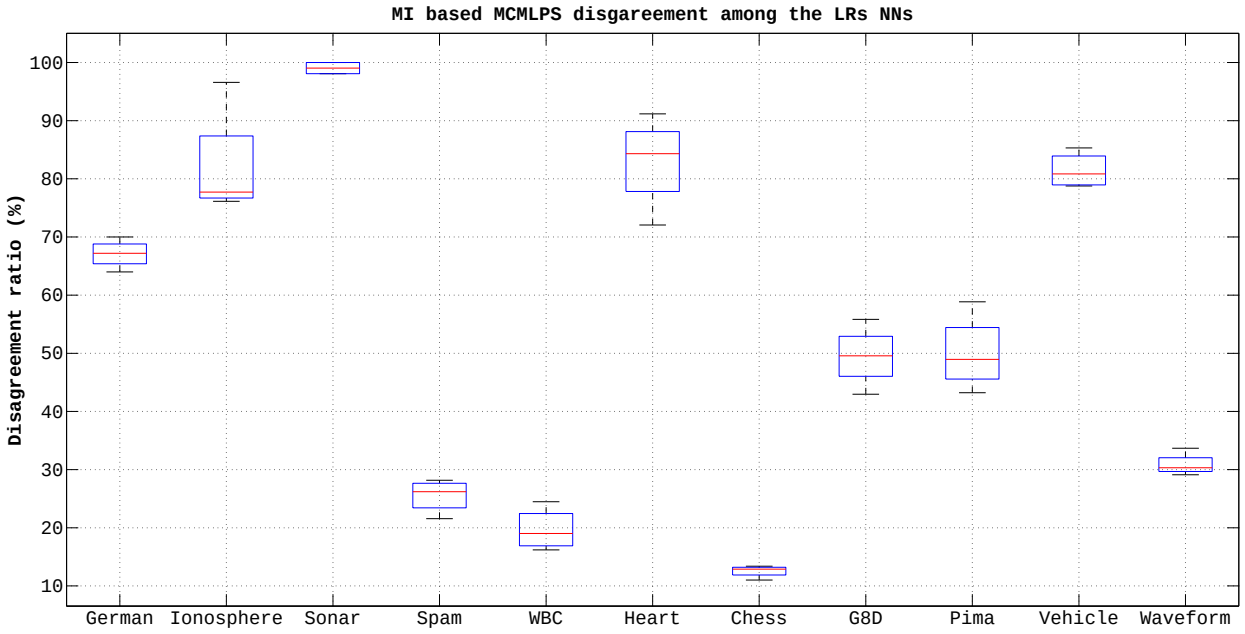


Figure 4.7: *Comparing the disagreements among the LRs of MI based MCMLPS when feedforward NNs are used as the base predictors.*

4.6.2.4 Variation of the conditional mutual information

This section investigates the effect of changing three aspect of the proposed MI based architecture. These are: modifying the equation used to find the LR seeds, partitioning the data using CV instead of DPS and changing the ratio of features located to the LRs.

4.6.2.5 Ignoring the inner correlation with respect to the class

In this case the conditional mutual information term $I(X_k, S|Y)$ is removed from equation 4.7. This transforms the feature selection process to mutual information feature selection proposed by Battiti (1994) given in the following equation:

$$J_{cmi}(X_k) = I(X_k; Y) - \beta I(X_k; X_j) \tag{4.9}$$

where β is a configurable parameter which Battiti suggest that the optimal value for it is often 1. Figure 4.8 shows the results of this case compared to the results of our proposed architecture. The aim of this section is to compare the case where correlated features are considered as redundant and are removed from the feature selection process with the case

where the inner correlation between the feature is assessed with respect to the class. It can be seen from the results that, apart from the ionosphere data set, the case where the inner correlation is considered in selecting the features, performs better than or as good as the case where the inner correlation is not considered in the selection.

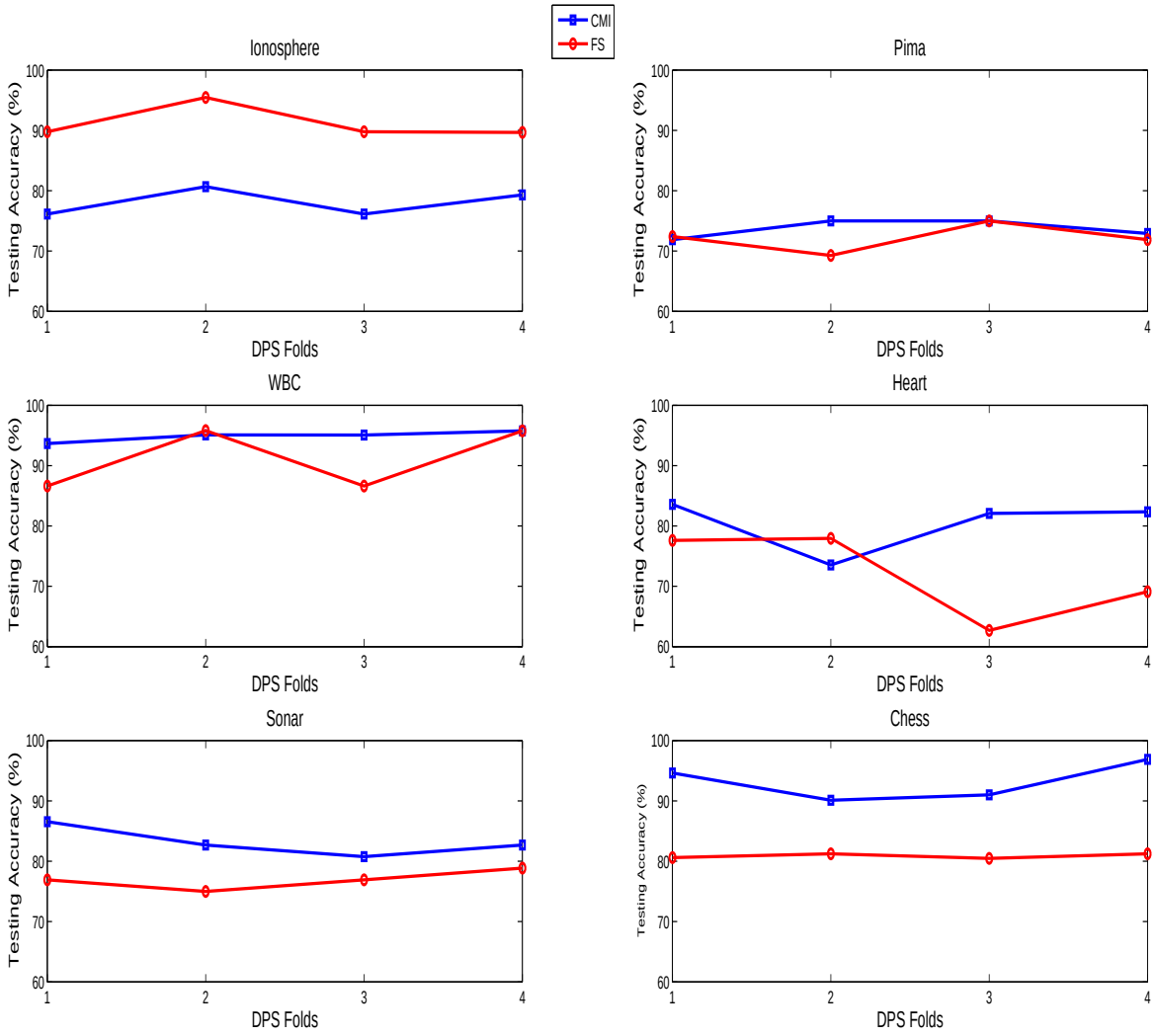


Figure 4.8: *Comparing the accuracy of the system when the data is split using CMI and traditional feature selection , part I.*

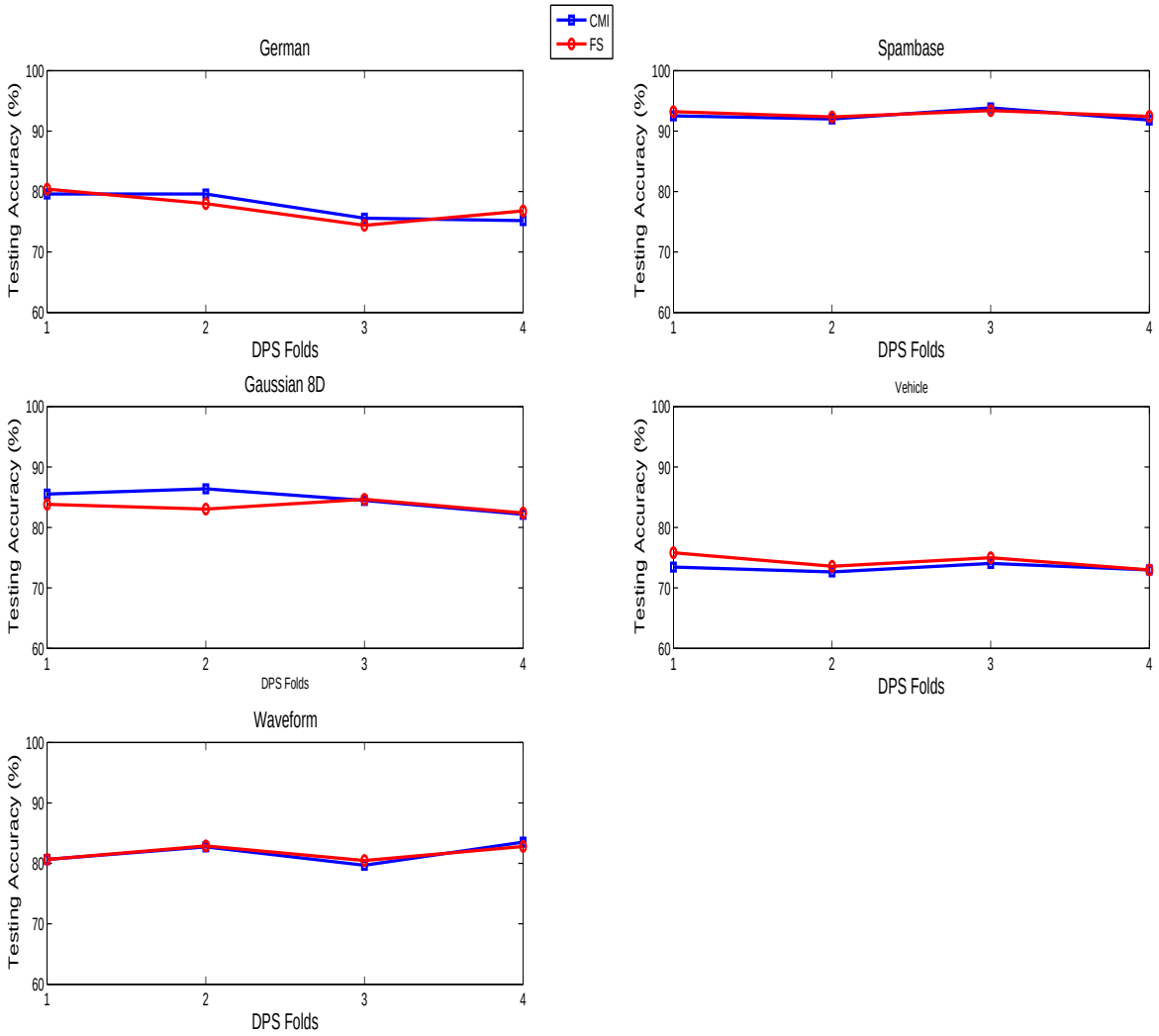


Figure 4.8: *Comparing the accuracy of the system when the data is split using CMI and traditional feature selection , part II*

4.6.2.6 Using Cross Validation instead of DPS

In this section the data of the MI based MCMLPS is partitioned using stratified CV instead of DPS. The CV is used to split the data into training and testing sets and to split the LRs data into K folds. Figure 4.9 compares the results of the two cases. It can be seen, that the two partitioning processes have comparable results, except for the ionosphere data set where CV performs better and the vehicle data set where the DPS performs better. The variation in the accuracies when CV is used with MI based MCMLPS is

much lower than the variation with the correlation based MCMLPS. This is due to the use of disjoint subset of the data to generate the LRs of the correlation based MCMLPS.

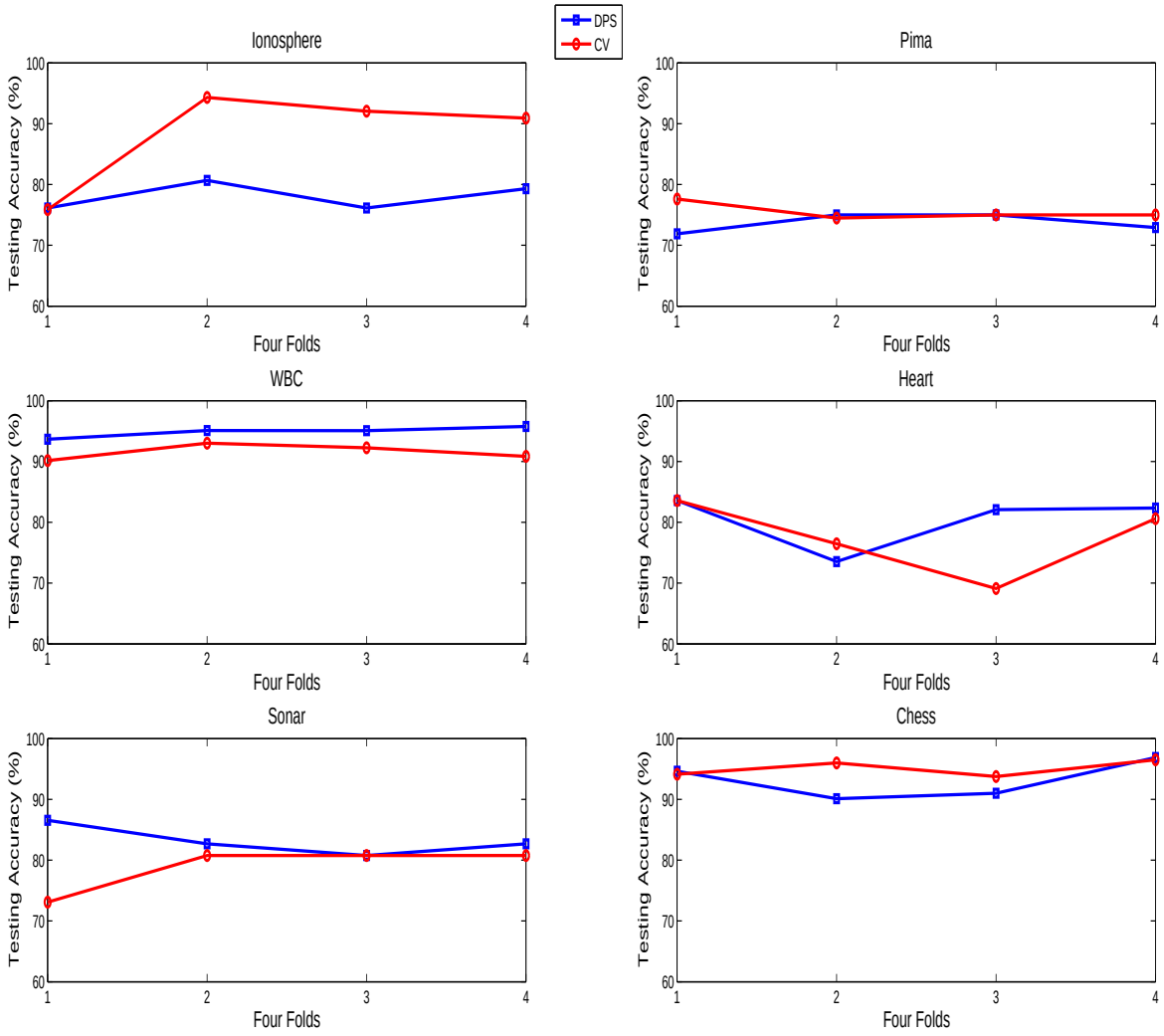


Figure 4.9: *Comparing the accuracy of the system when the data is split using CV and DPS, part I.*

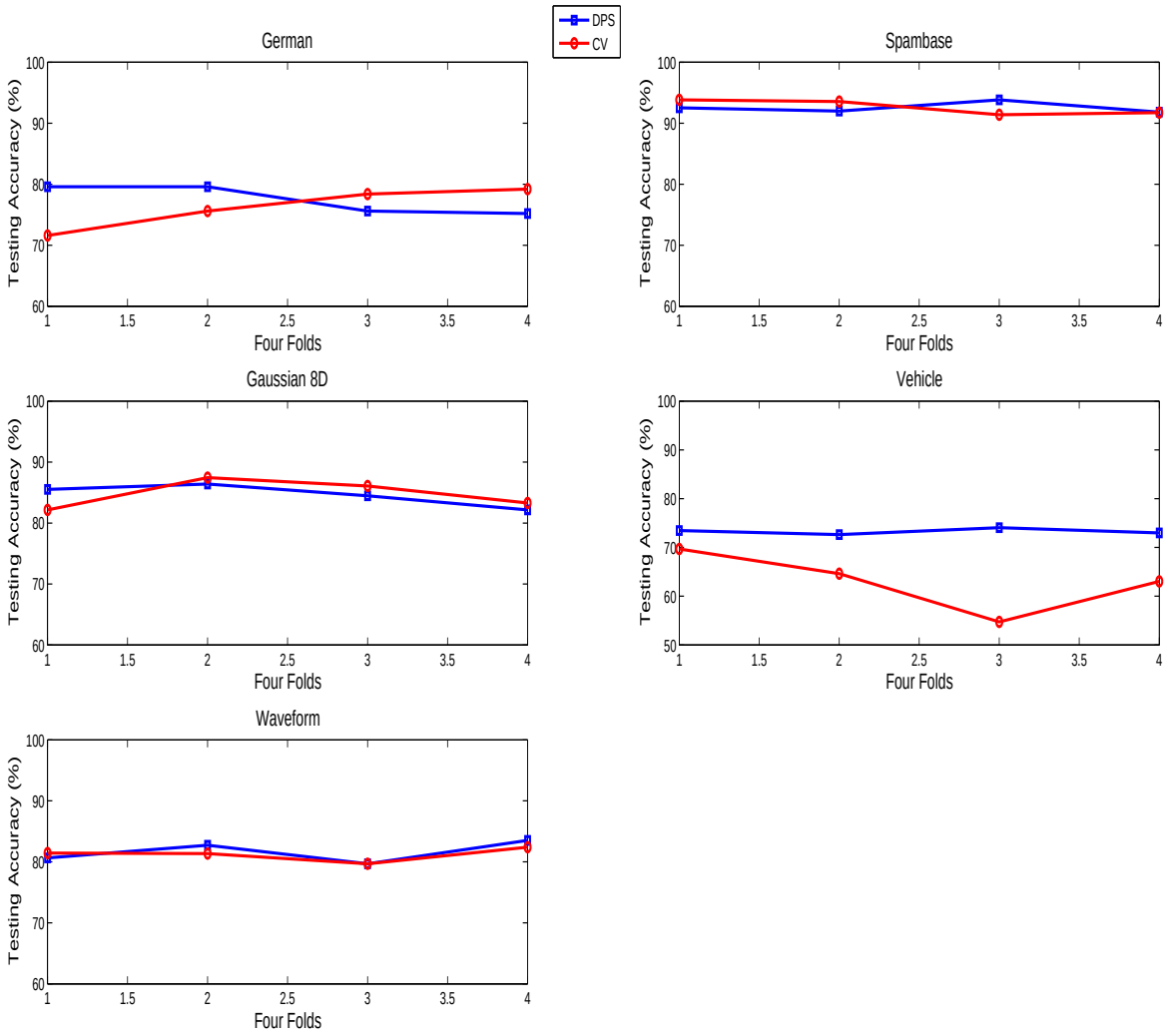


Figure 4.9: *Comparing the accuracy of the system when the data is split using CV and DPS, part II*

4.6.2.7 Changing the ratio of the features used in the LRs

In the previous experiments the ratio of features used in the LRs of the MI based MCMLPS was set to 30%. Using a higher or lower feature ratio have been tested on the data sets used in these experiments. It has been found that lowering this ratio from 30% to 10% decreases the accuracy of the LRs prediction as well as the overall accuracy of the system. On the other hand, increasing it to 80% result in a slight improving in the prediction accuracy for some of the data sets used in this experiment and it remained

unchanged for the rest. The only exception is for the ionosphere data set where the accuracy was increased to 92.30%. However, for some data sets the accuracy starts to drop after exceeding a certain threshold.

4.7 Summary

This chapter introduces a local learning based algorithm for multi-component, multi-layer architecture. This system divides the data into multiple LRs using the similarity of the features. Inside each LR a pre-defined number of base models are trained on subsets of data and/or subsets of features. The way in which the features are selected and assigned to the individual LRs depends on either the similarities of their pairwise squared correlation or their conditional mutual information. The squared correlation method can be applied in supervised as well as unsupervised learning as it does not consider the output class when splitting the data. On the other hand, the conditional mutual information method can be applied only in supervised learning as it uses the output class while splitting the data.

Investigating the internal performance of the proposed architecture (using either of the similarity metrics) showed that the overall testing accuracies of the architecture exceeded the average internal accuracies of its LRs models. This is due to the LRs being trained on either disjoint sets of data or subsets of features. However, since the prediction of the LRs is weighted by the similarity of the features to the seeds of the LRs, a higher degree of importance is given to the prediction of LRs that are most similar to the new data instance.

In the proposed architecture, the amount of variation in the internal accuracy depends mainly on the size and dimensionality of the data. Given an adequate amount of data used to train and validate the LRs models, the variation becomes small. Otherwise, the variation will be high. The results showed that both the number of LRs and number of models developed within the LRs need to be optimised with respect to the data set size and dimensionality.

The high level of complexity in the proposed MCMLPS is due to the use of multiple base models and to the procedure followed to generate the LRs. Nevertheless, it has a comparable performance to the benchmark algorithms. Despite that, the locality and the high level of diversity among the base predictors of the proposed architecture can be

beneficial in noisy environments. For example, when the noise is applied to only a part of the data, it will not have the same effect on all of the MCMLPS base predictors. The robustness of the proposed architecture to external noise will be investigated in the next Chapter.

Chapter 5

Multi-Component, Multi-Layer Predictive System in Noisy Environments

5.1 Introduction

This Chapter studies the relation between accuracy, diversity and robustness of the proposed MCMLPS in noisy environments. In ensemble learning, in order to improve the accuracy of the prediction, a number of factors have been studied in literature. These factors include: classifier selection (Zhang and Zhang (2009), Ko et al. (2008) and Parvin et al. (2011)), feature selection (Zhang and Zhang (2009), Zhang and Yang (2008) and Freund and Schapire (1996)), diversity creation in ensembles (Kuncheva et al. (2002), Hatami and Ebrahimpour (2007) and Kuncheva and Whitaker (2003)), fusion methods (Zhang and Zhang (2009), Hatami and Ebrahimpour (2007) and Al-Ani and Deriche (2002)) and combining more than one ensemble (Kotsiantis (2011), Panov and Dzeroski (2007) and Kotsiantis and Pintelas (2004)).

Some of these factors have been addressed in Chapter 4, where the proposed MCMLPS considered feature selection through the use of correlation based and mutual information based local features selection. The diversity among the base predictors was encouraged by training the models on subsets of the data and/or the features. Also, in the proposed system multiple ensembles were combined to obtain the final prediction.

In this Chapter, in addition to the previously considered factors, the effect of model se-

lection and of using different combiners on the performance of the proposed system is studied through the introduction of six fusion methods. Chapter 5 examines the robustness of the proposed system in practice. Both correlation based and MI based MCMLPS are tested on data sets with different ratios of noise added to either the training or the testing data. The performances of both systems are compared to three well known ensemble methods (namely, Bagging, Boosting and rotation forest).
The organization of Chapter 5 is as follows: in Section 5.2, different types of noise are explained and their effect on the prediction of machine learning methods is examined. Section 5.3 discusses balancing the robustness and the flexibility of machine learning methods. Section 5.4 examines the performances of both correlation based MCMLPS and MI based MCMLPS in noisy environments and compares their results to benchmark algorithms. In Section 5.5, six fusion methods are employed to combine the prediction of the base predictors/ensembles. Furthermore, the effect of using these combiners on the overall performance of the system is examined. Finally, Section 5.6 provides a summary for the Chapter.

5.2 The effect of noise on system prediction

To build a prediction model that can generalize well on new data, the following two factors should be considered (Hickey (1996)): a) the quality of the training data; and b) the inductive bias of the learning method. The quality of real world data is often affected by noise which can influence the performance of the learned model. According to (Hickey (1996)), noise can be defined as any variable that distorts the relation between the feature of an instance and its class. Adding noise to the data can result in a number of drawbacks (Hickey (1996) and Frénay and Verleysen (2014)) such as: reducing the prediction accuracy, increasing the amount of data and the time required to build the predictive model, and increasing the model complexity.
The quality of the data can be influenced by two factors (Zhu and Wu (2004)): a) an internal factor; and b) an external factor. The internal factor is related to the choice of the classes and the features and how they are defined to represent the underlying problem, while, the external factor is concerned with the error introduced in the classes and the features. Taking both factors into consideration there are three sources of physical noise:

- Insufficient description of the features and the classes.

- Corruption of features in the training examples.
- Erroneous classification of the training examples.

In real world data sets, the first source is hard to evaluate, as it is related to selecting the proper features which define the problem and assigning the classes. Meanwhile, the remaining two sources can be investigated (Zhu and Wu (2004)).

Noise in the data can be classified into two types: feature (attribute) noise and class (label) noise. Feature noise is related to the error introduced to the feature values. Examples of this type of noise include: erroneous feature values, missing feature values or incomplete features. An example of missing or incomplete features can be found in medical data, where the values for some medical tests might be unavailable for certain patients.

On the other hand, class noise is related to the noise that affects the class of an instance. According to Zhu and Wu (2004), there are two possible sources of class noise: contradictory examples, where the same instance is classified into different classes, and misclassifications where the instances are assigned to the wrong class. In literature, it has been shown that class noise is more harmful than feature noise (Zhu and Wu (2004)). This is due to the high impact of the class compared to the features.

This Chapter studies the effect of adding both class noise and feature noise to the data on the performance of the proposed MCMLPS. The main aspects of the proposed MCMLPS which will be investigated include:

- The robustness of the MCMLPS to noise compared to benchmark ensemble methods.
- The effect of assigning the LRs data using correlation based and MI based approaches on the accuracy and the stability of the system in the presence of noise.
- The effect of changing the weights and the fusion methods of the MCMLPS on the accuracy and the stability of the system in the presence of noise.
- The effect of training noise compared to testing noise on the performance of the proposed system.

The next section discusses two important concepts in designing predictive systems, these are: the robustness to noise and the flexibility of the system.

5.3 Balancing Robustness and Flexibility

The robustness of machine learning models refer to the model ability to maintain similar performance when it is tested on data similar to the training sample (Xu et al. (2009)).
The robustness of a learning algorithm is often conflicted with its flexibility (Hernández-Lobato (2010)). Since learning algorithms need to be flexible enough to capture the actual pattern in the data, yet they should be robust to outliers and noise. In machine learning literature, this concept is often discussed in terms of the bias-variance decomposition (Geman et al. (1992)). Flexible methods tend to have high variance and low bias, while robust methods tend to have high bias and low variance.
Bias-variance decomposition is often used to control the model complexity such that the complexity of a chosen model matches the complexity of the predicted problem. The complexity of the developed model can have direct effect on its generalization ability (Hastie et al. (2002)). Simple models tend to under-fit the data and thus can have poor generalization ability. Meanwhile, complex models tend to over-fit the data (by including noise and outliers) which can also lead to poor generalization ability. Generally, machine learning methods can be classified according to the complexity of the developed models into parametric methods (less complex) and non-parametric methods (more complex) (Alpaydin (2014)).
The parametric methods develop models that describe the data using fixed number of parameters (Wasserman (2013)). Due to the use of a predefined number of parameters, these methods make strong assumptions about the underlying function that generates the data. If the assumed function does not represent the actual function, in this scenario no matter how big is the available data, the parametric method will perform poorly. The use of parametric methods can be linked to Occam's razor principle which stated that simple models should be preferred, if using more complex models does not improve the quality of the prediction (Duda et al. (2012) and Domingos (2000)).
In general, parametric methods are robust, less expressive and less flexible than non-parametric methods (Wasserman (2013)). Due to their strong assumptions on the underlying function and the fixed number of parameters, they are less reliable in learning complex data patterns. Examples of parametric methods include: linear regression, Kalman filter (Kalman (1960)) and Markov random field model (Kindermann and Snell (1980)) among others.
On the other hand, in non-parametric methods the number of parameters is not prede-

fined. In addition, these methods make as few assumptions about the underlying function that generated the data as possible (Wasserman (2006)). As nonparametric methods do not make strong assumptions about the data, they can learn complex pattern with as much precision as desired. This property makes them more subjected to overfitting. Depending on the complexity of the learned pattern and the size of the data, the number of the model parameters increases. Examples of non-parametric methods include: neural networks (Bishop (1995)), decision trees (Breiman (1996)) and support vector machine (Vapnik (2013)) among others.

Parametric methods are often used with moderate or small size data that has a moderate level of noise. Furthermore, they can be used when a prior knowledge of the data pattern is known or the pattern has a simple form (Hernández-Lobato (2010)). In these cases the robustness of the model is more important than its flexibility, and parametric methods can have good performance. On the other hand, non-parametric methods are more suitable to large data sets with little or no knowledge about their distribution (Hernández-Lobato (2010)). In these cases more flexible models (non-parametric methods) are preferred over robust methods (parametric methods).

The proposed MCMLPS predictive system can use either parametric or non-parametric base predictors, however in its current setting it uses non-parametric base predictors (decision trees and neural network). Nevertheless, controlling the size and dimensionality of the training data as well as applying the pruning process to the decision trees allow certain control over the base model complexity.

5.4 Testing the MCMLPS in noisy environments

Traditionally, combining multiple classifiers has been linked to the ability of the system to perform accurately with noisy data (Ho et al. (1994) and Sáez et al. (2013)). One of the main motivations for associating robustness to noise with multiple classifiers systems is linked to the diversity among the system base models. Combining diverse models can improve the generalization ability of the system due to their complementary behaviour and allows the system to be less subjected to overfitting noisy data (Teng (1999) and Sáez et al. (2013)).

In this section the robustness of the MCMLPS proposed in Chapter 4 is tested in noisy environment. Furthermore, this section studies the effect of having highly diverse base

predictors (in correlation based MCMLPS) as well as having less diverse but more stable base predictors (in MI based MCMLPS) on the performance of the system in terms of its accuracy and robustness. The results of both systems are compared to the three benchmark ensemble methods, namely: bagging, AdaBoost and rotational forest.
As has been discussed in Section 5.2, the noise is often added to either the class or the features of classification problems. In the experiment presented in this Section, the noise is added to both the features and the classes in either training or testing sets. The aim of adding the noise to the testing data is to investigate the capability of the system to deal with noisy testing set when it has been trained on clean data. On the other hand, in the case of adding noise to the training data, the robustness of the system is investigated when the system is trained on noisy data and is required to provide prediction on clean data.
In this experiment, the noise is generated by taking the minimum and maximum values of each feature and producing random values within these limits. A random class (chosen from the set of problem classes) is generated and assigned to the new instances.
The following Subsection examines the performance of the proposed MCMLPS and the benchmark ensemble methods when applied to data with noise added to both the training and the testing data.

5.4.1 Results

This subsection compares the testing accuracies and the robustness of the correlation based MCMLPS and the MI based MCMLPS with the testing accuracies and the robustness of the three ensemble methods when applied to noisy data sets. The combiner method used in the MCMLPS is weighted majority vote, where the weights are the similarity values between the LRs data and the testing data. The calculation of the weights had been explained in Section 4.6.
Five ratios of noise are added to each data set, these ratios are: 0% where no noise is added, 10% noise added to the training data, 20% added to the training data, 10% added to the testing data, and 20% added to the testing data. The data sets used in this experiment are the same data sets that have been used in Chapter 4 (given in Table 4.3).
The results of the accuracy for the five benchmark algorithms are illustrated in Figures 5.1-5.11. Given below is a summary for the performance of each method:

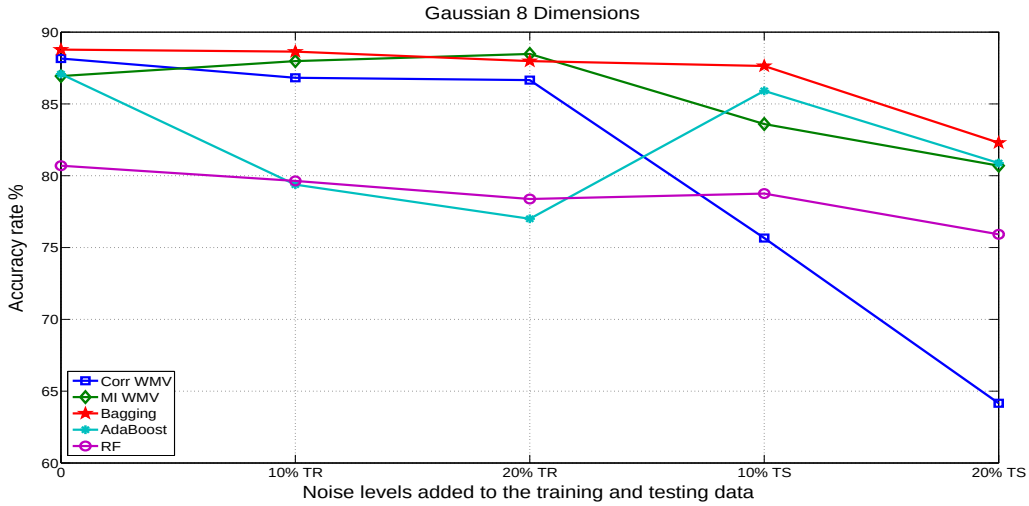


Figure 5.1: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the Gaussian 8D data set.*

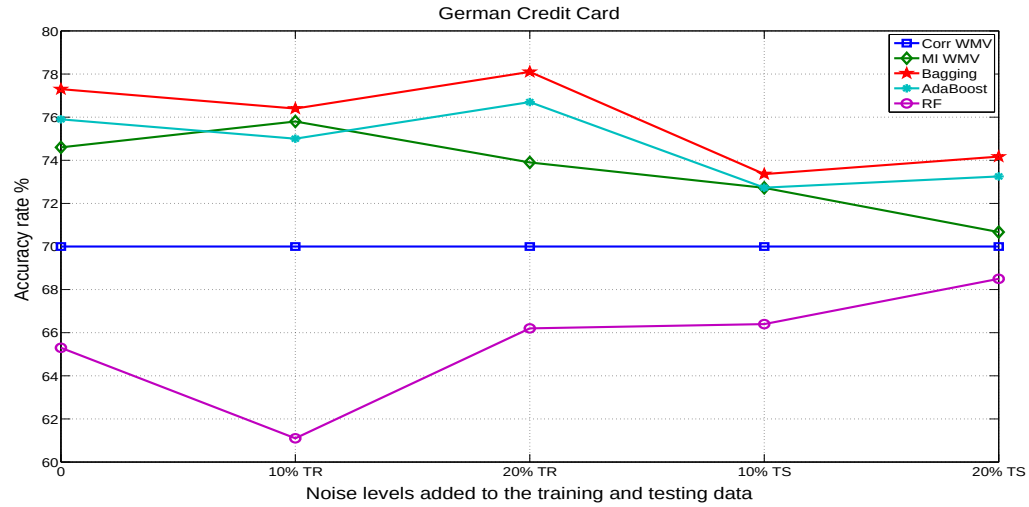


Figure 5.2: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the German credit card data set.*

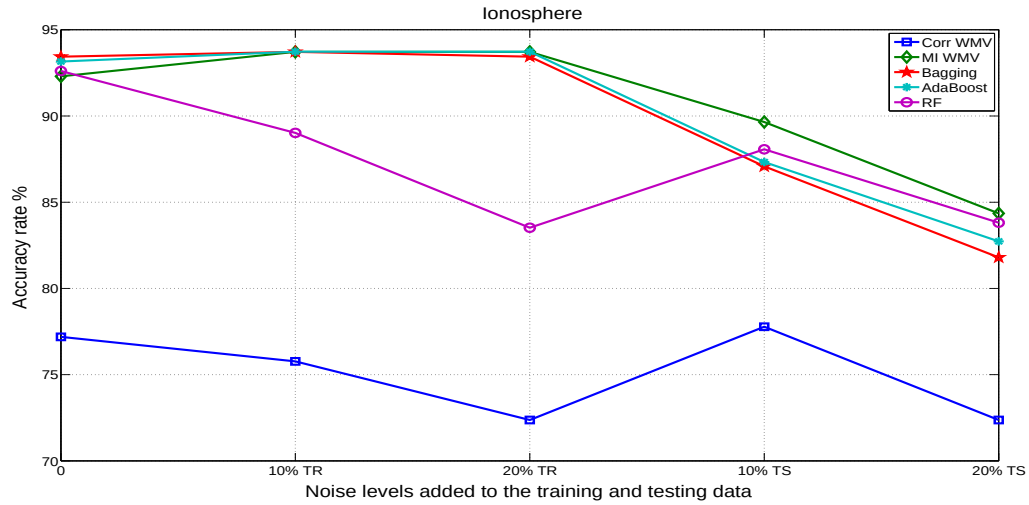


Figure 5.3: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the ionosphere data set.*

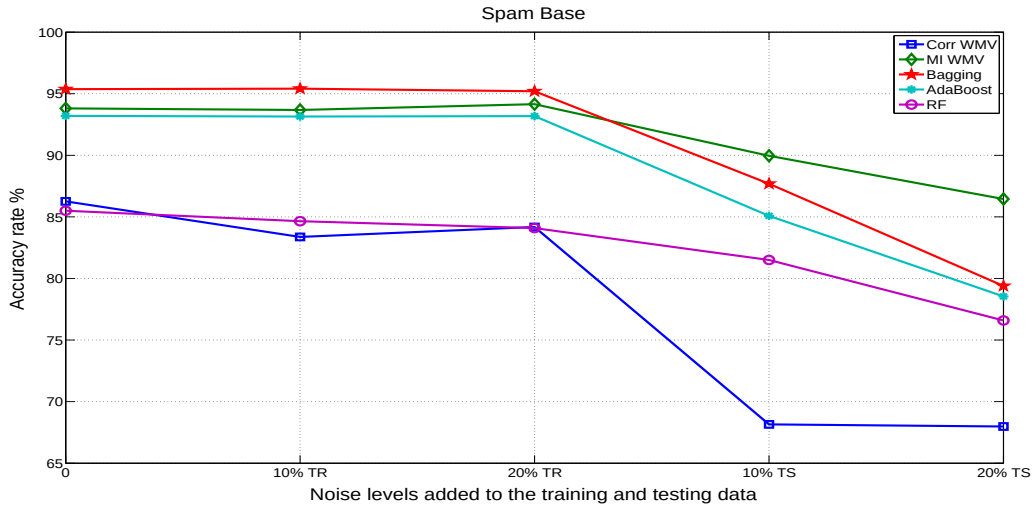


Figure 5.4: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the spam base data set.*

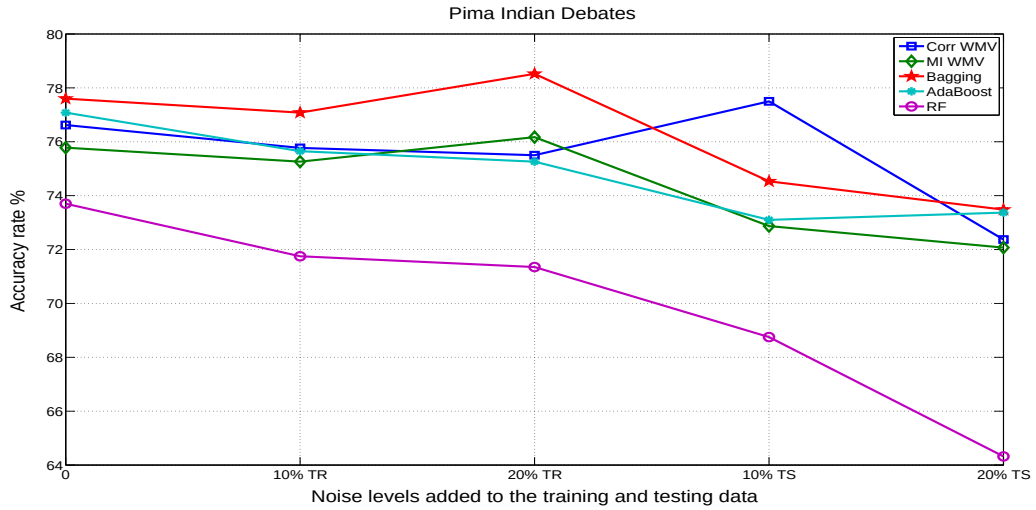


Figure 5.5: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the Pima Indian diabetes data set.*

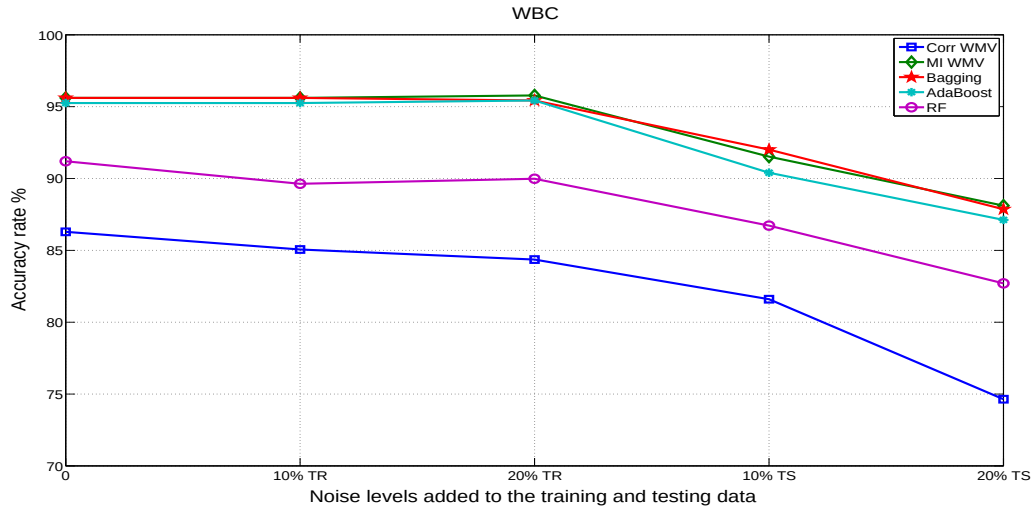


Figure 5.6: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the WBC data set.*

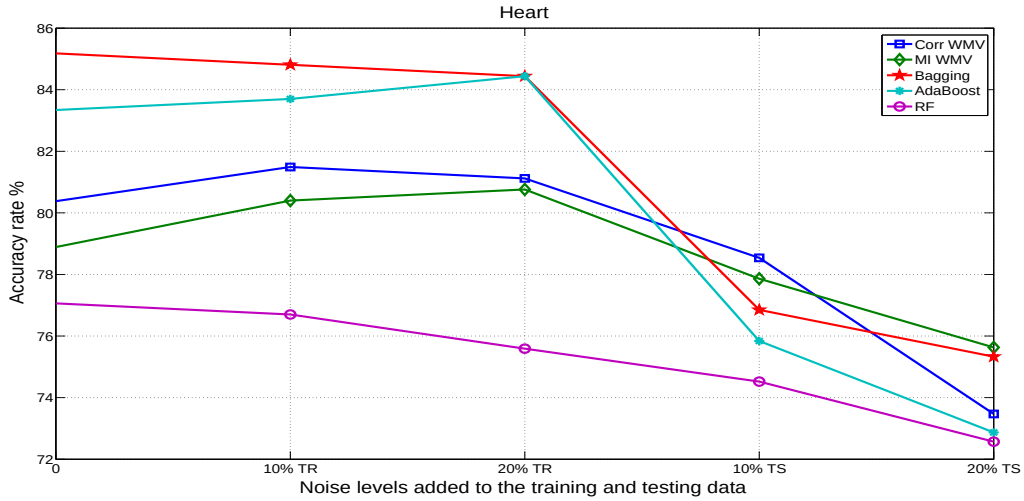


Figure 5.7: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the heart data set.*

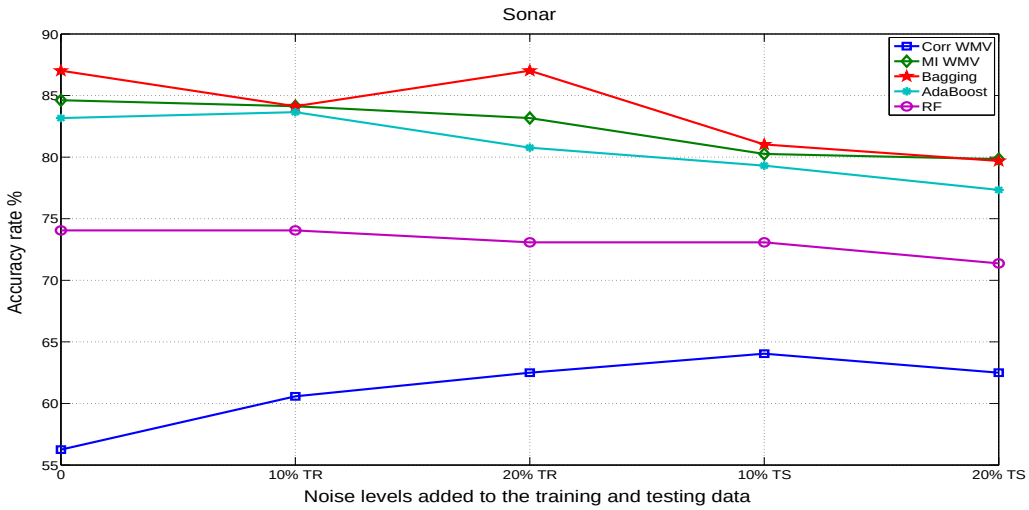


Figure 5.8: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the sonar data set.*

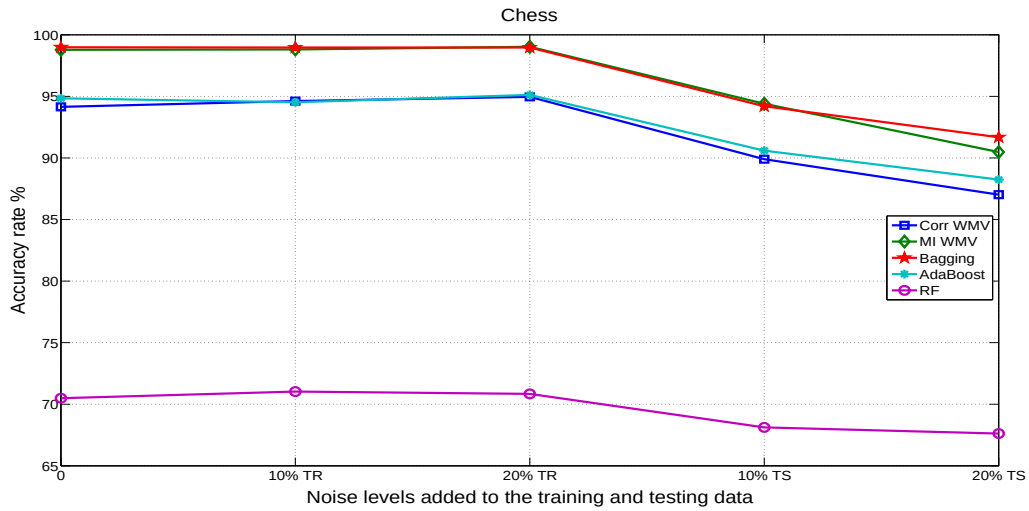


Figure 5.9: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the chess data set.*

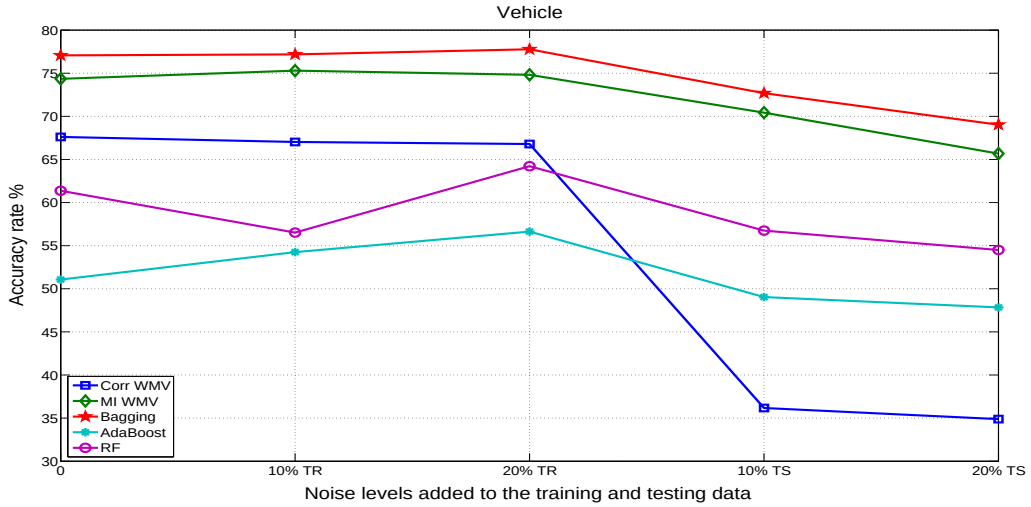


Figure 5.10: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the vehicle data set.*

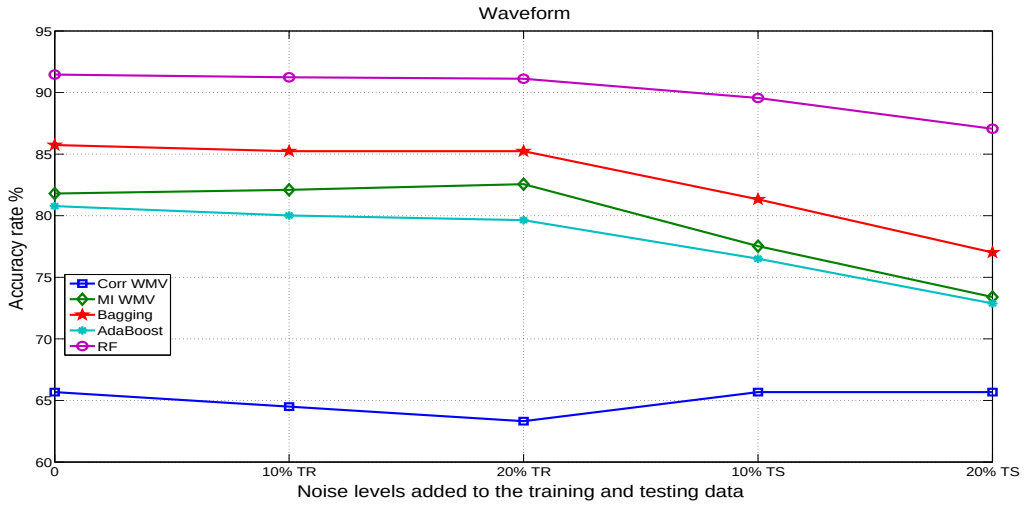


Figure 5.11: *Comparing the accuracy of the five benchmark algorithms in noisy environments for the waveform data set.*

- Bagging: this method has the highest accuracy for most of the data sets, the only exception is the waveform data set (shown in Figure 5.11) where it has a lower accuracy than RF.
- MI based MCMLPS: this method often has the second highest accuracy. In only two data sets: the heart data set (shown in Figure 5.7), and Pima data set (shown in Figure 5.5), it has a slightly lower accuracy than correlation based MCMLPS.
- AdaBoost: in general this method's accuracy is slightly lower than the previous two methods, except for the vehicle data set (shown in Figure 5.10) where it has

the lowest accuracy among the five ensemble methods. However, when the noise is added to the testing data, its accuracy is higher than correlation based MCMLPS.

- Correlation based MCMLPS: using this method, the accuracy is often lower than MI based MCMLPS. Generally, the size of the data has a high impact on the accuracy of this method. This can be seen in small data sets such as WBC (in Figure 5.6), sonar (in Figure 5.8) and ionosphere (in Figure 5.3) where this method has the lowest accuracy among the five ensemble methods.
- RF: this method often has one of the lowest accuracies among the five ensemble methods. The only exception is for the waveform data set, where this method has the highest accuracy. Also with small data sets (WBC, sonar and ionosphere) its accuracy is better than the accuracy of correlation based MCMLPS.

The above points summarise the performance of the five ensemble methods in terms of their accuracies. In order to compare their robustness, two factors are taken into consideration, these factors are: a) the standard deviation of their accuracies over the five noise ratios added to the data and b) the relative loss of accuracy which calculates the loss in accuracy when certain level of noise is added to the data compared to the case when no noise is added to the data.

The standard deviations of the accuracies for the five algorithms are presented in Table 5.1. These values are used to quantify the amount of variations in the accuracies of the five ensembles when the noise is added to the data, i.e. how the algorithm performance is affected by the noise. To analyse the variance of the accuracy for the five ensemble methods when different noise ratios are added to the data, a one-way ANOVA test is used. All values of the standard deviations analysed in this section and subsequent sections are generally normal at $p > 0.01$ (according to a non-parametric Lilliefors text). The test showed that, in terms of the variation in the accuracies, the performance of the proposed architecture is comparable to the benchmark algorithms, where there is no significant difference between the methods means ($F = (4, 50) = 1.52$, $4P = 0.21$). However, comparing the results in Table 5.1 showed the following:

Though RF has low standard deviations over the five noise ratios (Mean $(M) = 2.67$), it does not perform as well as the remaining benchmark algorithms. The standard deviations of Bagging ($M = 3.80$), MI based MCMLPS ($M = 3.09$) and AdaBoost ($M = 3.73$) accuracies are comparable to each other over the 11 data sets, with the highest standard deviation being 7.09 for Bagging (for the spam data set), 4.08 for MI

based MCMLPS (for the vehicle data sets) and 6.64 for AdaBoost (for spam data set). The correlation based MCMLPS ($M = 5.17$) has varied levels of standard deviations that reach a maximum of 17.33 for the vehicle data sets and a minimum of 0 for the German credit card.

In addition, comparing the stability of the MI based MCMLPS performance with the best performing method (Bagging) showed that, MI based MCMLPS had a smaller overall variation in its accuracies (Standard Deviation $(SD) = 0.88$) than Bagging ($SD = 1.47$).

The use of the standard deviation in this study aimed to show if there are large variations

Table 5.1: *The standard deviations of the accuracy for the five ensemble methods when applied to data sets with five different noise ratios.*

Data sets	Correlation based MCMLPS	MI based MCMLPS	Bagging	AdaBoost	RF
Gaussian 8D	10.33	3.31	2.71	4.31	1.78
German credit card	0.00	1.95	2.03	1.70	2.74
Ionosphere	2.59	3.95	5.30	4.94	3.81
Spam base	9.12	3.35	7.09	6.64	3.61
Pima	1.94	1.84	2.14	1.66	3.62
WBC	4.66	3.42	3.41	3.77	3.41
Heart	3.29	2.08	4.81	5.31	1.82
Sonar	3.01	2.22	3.37	2.64	1.10
Chess	3.51	3.79	3.42	3.09	1.62
Vehicle	17.33	4.08	3.79	3.65	3.99
Waveform	1.06	3.95	3.74	3.28	1.85

in the accuracies of the ensemble methods when different levels of noise are added to the data. However, it does not show the actual amount of loss in the accuracy or whether adding noise to the testing data or to the training data has a higher impact on the accuracy of the system. In order to examine these two aspects of the ensemble methods, the Relative Loss of Accuracy (RLA) is calculated. This metric has been used in a previous study (Sáez et al. (2013)) to measure the robustness of classification methods. The RLA is measured using equation 5.1 given below.

$$RLA_{x\%noise} = \frac{Acc_{0\%noise} - Acc_{x\%noise}}{Acc_{0\%noise}} \tag{5.1}$$

Where $RLA_{x\%noise}$ is the relative loss of accuracy when $x\%$ noise is added to the data, $Acc_{0\%noise}$ is the accuracy when no noise is added to the data and $Acc_{x\%noise}$ is the accuracy when $x\%$ noise is added to the data. Table 5.2 shows the RLA when 10% and

20% noise are added to either the testing or the training data. The smaller the value of this metric, the lower the loss in the accuracy of the ensemble method and the more robust the method is to noise.

Table 5.2: *The relative loss in accuracy for the five ensemble methods when noise is added to the training (TR) and testing (TS) data*

Noise ratio	Correlation based MCMLPS	MI based MCMLPS	Bagging	AdaBoost	RF
Gaussian 8 dimensional					
10%TR	0.015	−0.012	0.002	0.088	0.013
20%TR	0.017	−0.018	0.009	0.116	0.029
10%TS	0.142	0.038	0.013	0.013	0.024
20%TS	0.272	0.072	0.073	0.071	0.059
German credit card					
10%TR	0	-0.017	0.012	0.012	0.064
20%TR	0	0.009	-0.010	-0.012	-0.014
10%TS	0	0.025	0.051	0.042	-0.017
20%TS	0	0.053	0.041	0.035	-0.049
Ionosphere					
10%TR	0.018	-0.0153	-0.003	-0.006	0.039
20%TR	0.062	-0.015	0	-0.006	0.098
10%TS	-0.008	0.029	0.068	0.063	0.049
20%TS	0.062	0.086	0.125	0.112	0.095
Spam					
10%TR	0.034	0.001	-0.0001	0.001	0.010
20%TR	0.024	-0.004	0.002	0.001	0.017
10%TS	0.210	0.041	0.081	0.087	0.047
20%TS	0.212	0.078	0.168	0.157	0.104
Pima					
10%TR	0.011	0.007	0.007	0.019	0.026
20%TR	0.015	-0.005	-0.012	0.024	0.032
10%TS	-0.012	0.038	0.040	0.052	0.067
20%TS	0.056	0.049	0.053	0.048	0.127

WBC					
10%TR	0.014	0	0	0	0.017
20%TR	0.022	-0.002	0.002	-0.002	0.013
10%TS	0.054	0.043	0.038	0.051	0.049
20%TS	0.135	0.078	0.081	0.085	0.093
Heart					
10%TR	-0.014	-0.019	0.004	-0.004	0.005
20%TR	-0.009	-0.024	0.009	-0.013	0.019
10%TS	0.023	0.013	0.098	0.090	0.033
20%TS	0.086	0.041	0.116	0.126	0.058
Sonar					
10%TR	-0.077	0.006	0.033	-0.006	0
20%TR	-0.111	0.017	0	0.029	0.013
10%TS	-0.138	0.052	0.069	0.046	0.013
20%TS	-0.111	0.056	0.084	0.070	0.0362
Chess					
10%TR	-0.005	-0.0003	0.0002	0.003	-0.008
20%TR	-0.009	-0.003	0.0002	-0.003	-0.005
10%TS	0.045	0.044	0.048	0.045	0.034
20%TS	0.076	0.084	0.074	0.070	0.041
Vehicle					
10%TR	0.009	-0.012	-0.002	-0.062	0.079
20%TR	0.012	-0.006	-0.009	-0.109	-0.046
10%TS	0.465	0.053	0.057	0.040	0.075
20%TS	0.484	0.117	0.104	0.063	0.112
Waveform					
10%TR	0.018	-0.004	0.006	0.009	0.002
20%TR	0.036	-0.009	0.006	0.014	0.004
10%TS	0	0.052	0.051	0.053	0.021
20%TS	0	0.103	0.102	0.098	0.048

The results in Table 5.2 showed that, though Bagging had the highest accuracy for most of the data sets, it often has a higher loss in accuracy compared to the best performing

method. On the other hand, Table 5.2 investigated 44 cases of added noise for the 11 data sets (4 noise ratios added to each data set), the results showed that the MI based MCMLPS had a lower loss of accuracy for 32 cases compared to Bagging.
In addition, the table shows that generally adding noise to the testing data has a higher effect on the accuracy of Bagging, MI based MCMLPS and AdaBoost than when the noise is added to the training data. Meanwhile, whether the performance of the correlation based MCMLPS and the RF is more effected by the training noise or the testing noise, this depends mainly on the data sets used.

5.4.2 Discussion

This experiment has compared the robustness and accuracy of the correlation based and MI based MCMLPS proposed in Chapter 4 to other ensemble methods in noisy environments.
The results showed that in terms of the accuracy of the prediction, in most cases Bagging has the highest accuracy. Nevertheless, it has a small difference in accuracy to that of the MI based MCMLPS. On the other hand, correlation based MCMLPS has a lower accuracy than Bagging, MI based MCMLPS and AdaBoost for most data sets and especially for small data sets. This is due to the multiple splits of the data in the correlation based MCMLPS, which had led to the LRs models being trained on a limited amount of data. The only exceptions where correlation based MCMLPS had a relatively higher accuracy than MI based MCMLPS are in Pima and heart data sets. These data sets had not only small number of samples but also small number of features. Though MI based MCMLPS allows its LRs models to train on all the available data, it uses only 30% of the features to train the base predictors. Due to this, having low number of features and samples can affect the performance of this method.
Meanwhile, the robustness of the five ensemble methods was tested using both the standard deviation of their accuracies and the RLA. Comparing the standard deviations for the considered methods showed that Bagging, MI based MCMLPS and AdaBoost have a comparable amount of variations, though the highest amount of variation in MI based MCMLPS is lower than that in the other two ensemble methods. Moreover, the correlation based MCMLPS have a varied level of variations in its accuracy and it can reach very large values for some of the data sets compared to the other ensemble methods.
The results of the RLA showed that MI based MCMLPS have a lower loss of accuracy

than Bagging in most cases. Moreover, it showed that Bagging, MI based MCMLPS and AdaBoost are more affected by testing noise than by training noise. Also, as was explained earlier, whether the performance of the correlation based MCMLPS and the RF is more affected by the training noise or the testing noise, this depends mainly on the data sets used.
In general, though Bagging often has a slightly higher accuracy than MI based MCMLPS, MI based MCMLPS can often provide more robust performance in terms of the variation in accuracy and the RLA.

5.5 The Effect of Changing the Fusion Methods on MCMLPS Performance

In the previous Section Weighted Majority Vote (WMV) had been used to combine the MCMLPS base predictors. The weights were calculated using either the pairwise square correlation or the conditional mutual information of the features.
In this Section, the effect of changing the fusion method of the base predictors/ensembles on the overall performance of the MCMLPS is examined. Six fusion methods are introduced and tested in noisy as well as non-noisy environments. A description for these methods is given below:

1. Single LR: this fusion method allows only one LR to provide the final prediction of the system. The LR is selected using the similarity metric measured between the LR seed and the new data features, such that, the LR with the highest similarity value is chosen. The base predictors of this LR are combined using WMV, where the weights are the similarity values.
2. Best Model: only the model with the highest accuracy from each LR is selected and combined together using WMV. The models performance is evaluated using their training accuracies.
3. MV: this fusion method is the unweighted majority vote (Mazurov et al. (1987)) and it represents the case where the similarity is not taken into consideration in prediction. Equal weights are assigned to all base predictors and the final vote is obtained by choosing the class of the majority.
4. WMV with the similarity metric: this fusion method is a weighted majority vote

(Shapley and Grofman (1984)), where the weights are the summation of the similarities between the data instances and the LRs data.

5. WMV with the similarity metric and the training accuracy in the first layer: this fusion method weights the prediction of the base predictors using the similarity metric and their training accuracies. The training accuracy is included in weighting the prediction of the first layer predictors. It assigns a higher weight to the base predictors which are more accurate in their predictions during the training phase. The remaining layers are weighted using the similarity metric.
6. WMV with the similarity metric and the accuracy in all layers: in this fusion method, the training accuracies of the base predictors are included in calculating the weights used across all the ensemble layers. In the first layer, the prediction of each model is weighted by the similarity metric as well as the training accuracy of that model. Meanwhile, in the subsequent layers the average accuracy of the ensemble base models are used with the similarity metric to weight the prediction of the ensembles.

These six fusion methods are used with both the correlation based MCMLPS and the MI based MCMLPS. The resulted system is applied to the data sets given in Table 4.3 with different noise ratios added to either the training or the testing data. In the following Subsections, detailed descriptions for these six fusion methods are provided. Furthermore, the performances of the MI based and the correlation based MCMLPS using these fusion methods are evaluated and compared. Both the testing accuracy and the standard deviation of the accuracy over the five added noise ratios are discussed in Subsections 5.5.1-5.5.4. Finally Subsection 5.5.5 examines and compares the RLA for the fusion methods in both correlation based and MI based MCMLPS.
The complete results for the RLA and the accuracy of the six fusion methods are presented in Appendix B.

5.5.1 Single local region

In this method, a single LR ensemble is chosen to provide the final prediction of the MCMLPS system. This ensemble is chosen based on the similarity between the features of its training data and the features of the new data sample. The base predictors of the chosen LR are combined using WMV, where the weights are calculated using the similarity between the features of the LRs seeds and the features of the testing data.

The ensemble can be trained on disjoint subsets of the data (as in the correlation based MCMLPS) or it can be trained on a subset of the features (as in the MI based MCMLPS). Figure 5.12 illustrate the single LR fusion method (this architecture and its parameters were explained in Section 4.2).

In the correlation based MCMLPS, using this fusion method often produces the lowest

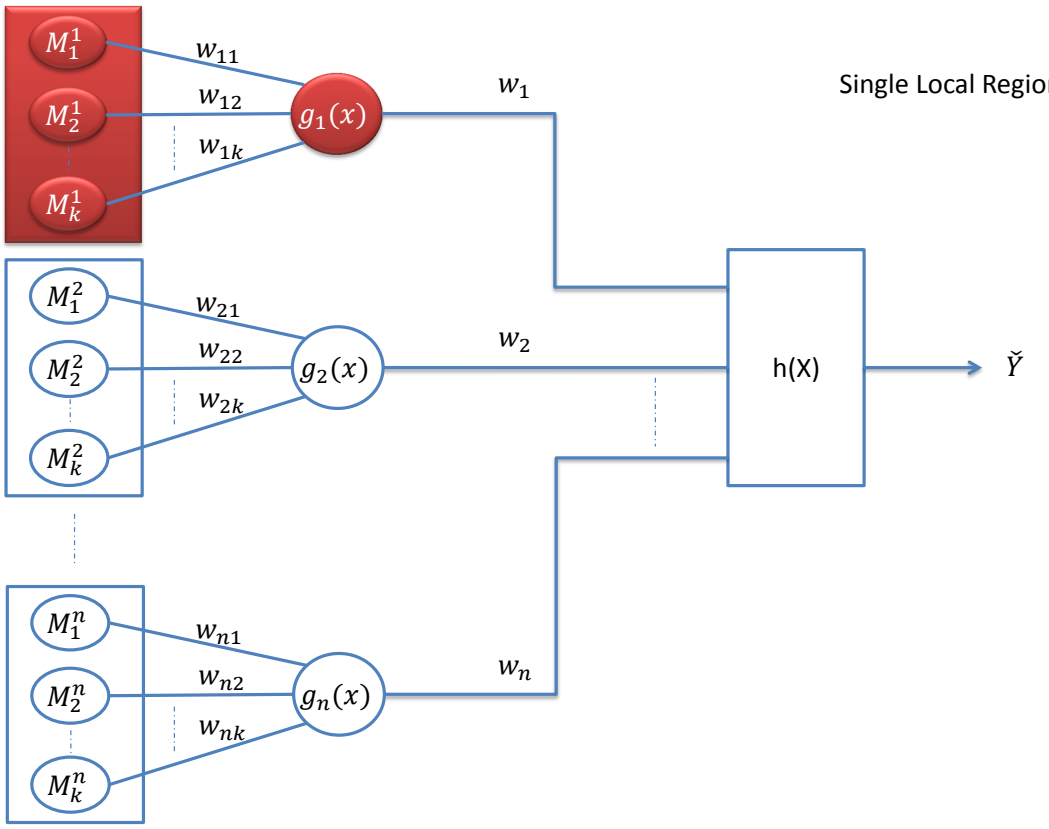


Figure 5.12: *An illustration for the MCMLPS with single LR fusion method*

accuracy among all six fusion methods. Except for the Gaussian 8 dimension data set and the German credit data set, where the performance of this method exceeds that of the MV. Table 5.7 (shown after Subsection 5.5.4) shows the accuracies of the single LR compared to the remaining fusion methods when the correlation based MCMLPS is applied to the Gaussian 8 dimension data set. This data set represents a binary classification problem with a reasonable number of samples. Thus single LR, having enough data to train on, is able to provide better accuracy than MV.

The reason for the single LR method to often have the lowest accuracy in correlation based MCMLPS is that, in this method the LRs ensembles are trained on disjoint subsets

of the data. This results in N independent ensembles. Though the similarity is taken into consideration when selecting the single LR, this ensemble is often trained on a small amount of data. Furthermore, a new sample can have two or more LRs which have a very close correlation values with the same sample. Due to this, the prediction using a single LR might not be as good as when multiple LRs ensembles are combined.

Meanwhile, in the MI based MCMLPS, this fusion method had a better performance compared to the correlation based MCMLPS. Choosing a single LR to provide the final prediction of the system often had better accuracy than MV and either better than or very comparable accuracy to the best model fusion method. Generally, the accuracy of the single LR is lower than WMV fusion methods. However, there are two exceptions to this case: the ionosphere data set (shown in Table 5.8) and the German credit card data set (shown in Table 5.14). In the ionosphere data set, single LR method has the highest accuracy alongside the best model method. Meanwhile in the German credit data set, it has a higher accuracy than WMV methods when there is no noise added to the data or when the noise is added to the testing data.

The standard deviation of the accuracies for the single LR fusion method using both correlation based and MI based MCMLPS is shown in Table 5.3. Applying a one way ANOVA test showed that there is no significant differences between the means of the standard deviation for the two systems ($F(1, 20) = 0.53$, $p = 0.47$). However, it can be noted that, using correlation based system can result in a high standard deviation value for certain data sets, such as, Gaussian 8 dimension, spam and vehicle data sets. Due to this the overall mean for the correlation based MCMLPS ($M = 4.09$) is higher than that for the MI based MCMLPS ($M = 3.11$).

Generally, though the accuracy has improved in MI based MCMLPS compared to correlation based system; the standard deviation of the accuracy for this system is slightly higher than that for the correlation based system for 7 out of the 11 data sets.

5.5.2 Best Model

The previous fusion method explores whether choosing a single LR ensemble can provide the best accuracy for the data sets using pairwise squared correlation and conditional MI similarity metrics. Though the performance is improved in the MI based MCMLPS, choosing a single LR often has lower accuracy than combining multiple components of the MCMLPS. This section explores if choosing the best base predictor from each LR

Table 5.3: *The standard deviations of the accuracy for single LR fusion method when applied with correlation based and MI based MCMLPS.*

Data sets	Correlation based MCMLPS	MI based MCMLPS
Gaussian 8D	9.19	2.63
German credit	0	1.73
Ionosphere	1.77	3.87
Spam	8.56	3.33
Pima	1.19	1.96
WBC	3.51	3.46
Heart	2.06	2.10
Sonar	1.40	3.05
Chess	3.21	3.76
Vehicle	13.54	4.38
Waveform	0.53	3.90

can provide better or as good performance as combining all of the base predictors. The aim is to show that if generating an ensemble using all the base predictors may result in combining redundant predictors or if they can convey different information about the problem.

This fusion method chooses the best model from each LR (based on the model training accuracy) and combines them in a single ensemble. The best models are combined using WMV method weighted by the similarity metric. Figure 5.13 illustrates this fusion method.

In the correlation based MCMLPS, generally this fusion method performs better than single LR fusion method and in most cases it is better than MV method, however, its performance is worse than WMV methods (as will be shown later). There are certain exceptions, for example, in the ionosphere data set shown in Table 5.9, it performs slightly better than WMV and has the second highest accuracy after MV. Furthermore, when a high ratio of noise (20%) is added to either the training or the testing data, this fusion method has the highest accuracy. Moreover, for the Pima data set it has the second highest accuracy after the MV fusion method.

On the other hand, the accuracy of this fusion method in the MI based MCMLPS is higher than in the correlation based MCMLPS. Compared to other fusion methods, the accuracy of the best model fusion method in the MI based MCMLPS is lower than WMV methods. It has a comparable accuracy to the single LR method and often is better than MV method. Some exceptions to this case include: the vehicle data set and the Pima data

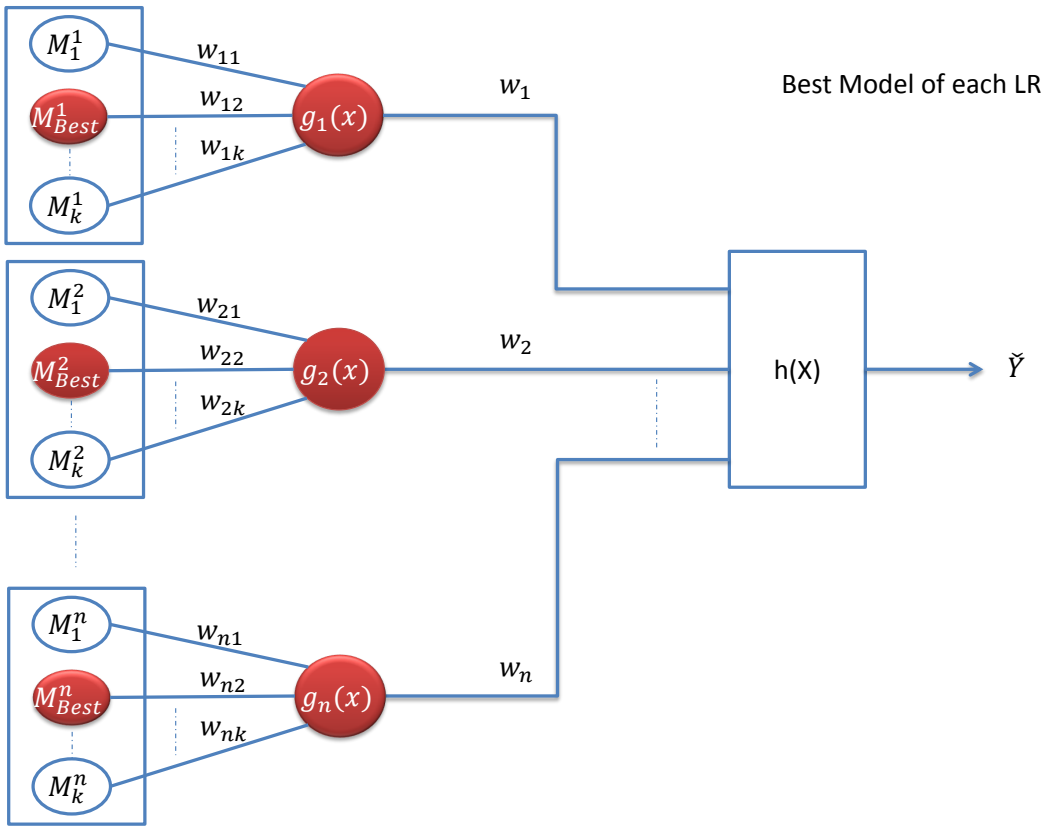


Figure 5.13: *An illustration for the MCMLPS with best model fusion method*

set. In the vehicle data set, this fusion method has a lower accuracy than MV method. Meanwhile, in the Pima data set choosing the best model has the lowest accuracy except when the noise is added to the training data, where it becomes better than MV and single LR. Table 5.10 shows the accuracies for the best model fusion method compared to the other fusion methods when MI based MCMLPS is applied to Pima data set.

The robustness of the best model fusion method is affected by the method used to train the base predictors. The standard deviations of the accuracies using this fusion method for the correlation based MCMLPS ($M = 4.59$) and the MI based MCMLP ($M = 3.41$) is shown in Table 5.4. Applying a one way ANOVA test showed that there is no significant differences between the means of the standard deviation of the two systems ($F(1, 20) = 0.64$, $p = 0.43$). The correlation based MCMLPS had a lower variation in the accuracy than MI based MCMLPS for 7 out of the 11 used in the experiments.

The standard deviation for the accuracy of the best model fusion method is comparable to that of the single LR fusion method. Furthermore, in the case of correlation based

MCMLPS, both fusion methods does have similar high values of standard deviations of the accuracy when applied to the Gaussian 8 dimension, spam and vehicle data sets.

Table 5.4: *The standard deviations of the accuracy for the best model fusion method when applied with correlation based and MI based MCMLPS.*

Data sets	Correlation based MCMLPS	MI based MCMLPS
Gaussian 8D	9.96	2.97
German credit	0	1.95
Ionosphere	1.29	3.65
Spam	10.77	3.28
Pima	1.23	4.07
WBC	3.85	3.81
Heart	2.06	4.33
Sonar	1.40	1.91
Chess	3.77	3.82
Vehicle	14.42	3.83
Waveform	1.76	3.90

5.5.3 Majority vote

The previous two fusion methods had tested for selecting parts of the MCMLPS to provide the prediction of the system and whether this procedure can result in a good performance for the system in terms of its accuracy and robustness. The MV fusion method combines all the base predictors to provide the final prediction for the system. It assigns equal weights for the base predictors without taking into consideration the similarities of the new sample to the LRs data. Figure 5.14 illustrates this fusion method. The aim of designing MV fusion method is to test for the significance of using the similarity and the training accuracy to weight the prediction of the system.

In the correlation based MCMLPS, in most cases, the accuracy of MV is lower than WMV. The exception for this is when data sets with small number of samples and/or features are used, such as: ionosphere, Pima and sonar data sets. Furthermore, since in correlation based MCMLPS the data is usually split into disjoint subsets with limited number of features, the LRs models are trained on small amount of data. Due to this, in small data sets using both the similarity and the training accuracy of the base model might not help in improving the accuracy of the correlation based MCMLPS. However,

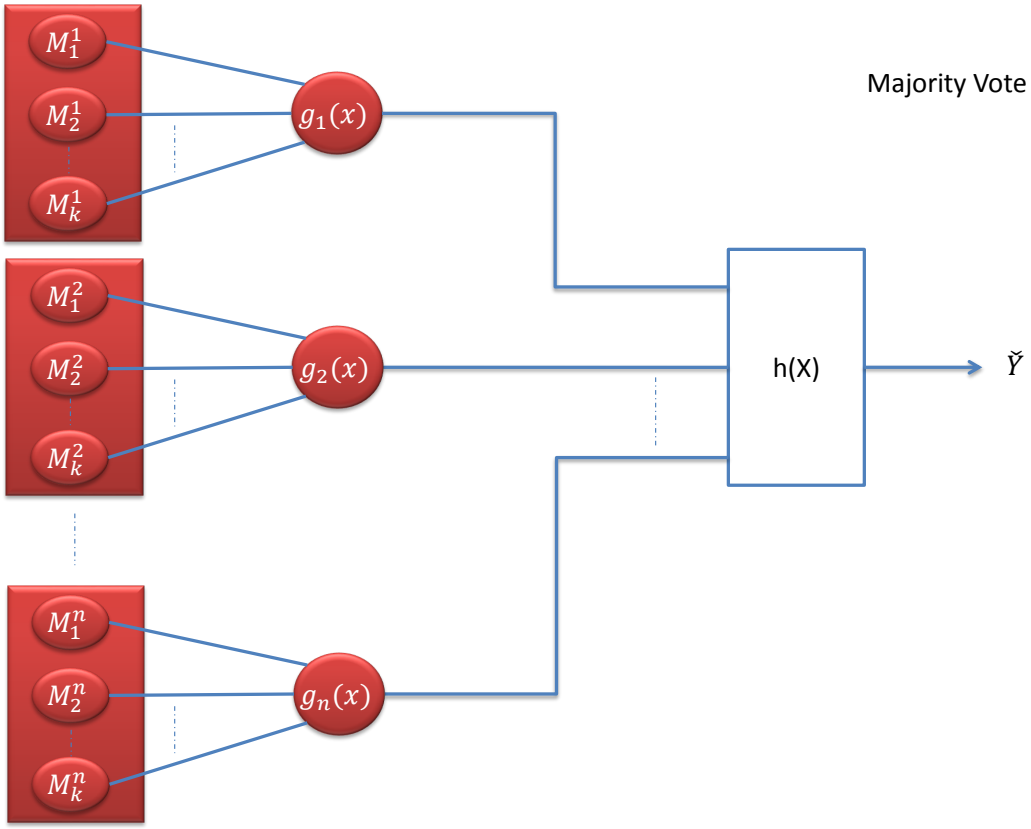


Figure 5.14: *An illustration for the MCMLPS with MV fusion method*

as the ratio of noise increases the differences in the accuracy between MV and WMV fusion methods decreases. Table 5.11 shows the accuracies of the MV fusion method compared to the remaining fusion methods when applied to the sonar data set. Generally, in correlation based MCMLPS, MV fusion method has better accuracy than choosing single LR to provide the final prediction of the system. In some data sets, especially in multi-class classification problems, it is better than the single LR and best model fusion methods.

On the other hand, in MI based MCMLPS, the accuracy of the MV fusion method is lower than WMV methods on all data sets, and the only exception is that for the vehicle data set (Shown in Table 5.12) where its accuracy is slightly better than WMV methods when the noise is added to the testing data. Generally, in MI based MCMLPS, MV fusion method has a lower accuracy compared to single LR and best model fusion methods.

The standard deviations of the accuracies of the MV fusion method, when applied to the 11 data sets, are shown in Table 5.5. Comparing the standard deviations of the accuracies

in the MI based MCMLPS ($M = 2.97$) and correlation based MCMLPS ($M = 5.35$) shows that, unlike the previous two fusion methods, in most data sets using the MV with MI based MCMLPS results in lower deviations in the accuracies than when it is used with correlation based MCMLPS. However, analysing the variance of the accuracies for both systems, using one-way ANOVA test, showed that there is no significant difference between their means ($F = (1, 20) = 2.18$, $p = 0.16$).

Table 5.5: *The standard deviations of the accuracy for the best model fusion method when applied with correlation based and MI based MCMLPS.*

Data sets	Correlation based MCMLPS	MI based MCMLPS
Gaussian 8D	8.24	2.95
German credit	1.36	1.24
Ionosphere	4.21	4.13
Spam	11.29	3.08
Pima	3.12	1.75
WBC	3.62	3.75
Heart	3.47	1.61
Sonar	0.62	2.98
Chess	3.52	3.71
Vehicle	18.02	3.66
Waveform	1.32	3.78

5.5.4 Weighted majority vote

This subsection discusses three WMV fusion methods. Similarly to MV, in these methods all of the base predictors are combined to provide the final prediction of the system. However, unlike the MV where equal weights are assigned to the predictors/ensembles, in these fusion methods three weighting vectors are considered. The first fusion method, illustrated in Figure 5.15, weights the prediction of the base predictors using the similarity metric used to split the data into LRs. As mentioned previously, two similarity metrics are used in the experiments, these are: the pairwise squared correlation metric and the conditional mutual information metric. The aim of weighting the base predictors/ensembles using one of the similarity metrics is to allow a higher degree of importance to the LRs that are most similar to the new data sets. Therefore, this fusion method tests for the accuracy and the robustness of the prediction when the locality of the new data is taken

into consideration.
The second fusion method, shown in Figure 5.16, uses both the normalised accuracy of the base predictors and the similarity metric to weight the prediction of the first layer of the system while only the similarity metric is used to weight subsequent layers. Including the accuracy in the weighting vector aims to associate how accurate the base predictors/ensembles are with the weights assigned to the LR. This can help the system not only to base its weights according to the locality of the data, but also it allows more accurate predictors/ensembles to have a higher impact on the prediction of the system.
The third fusion method, shown in Figure 5.17, uses the normalised accuracy and the similarity metric to weight the first layer. In the subsequent layers, it uses the similarity metric and the average accuracy of the base models/ensembles to weight the prediction.
This section will highlight how these fusion methods perform compared to each other as well as to the other previously discussed methods. Table 5.6 will refer to these three fusion methods as WMV 1, WMV 2 and WMV 3 respectively.
In Figures 5.15 to 5.17, $sim_1, ..., sim_n$ represent the similarities between the LRs data and the test data, $TR_Acc_1, ..., TR_Acc_k$ are the accuracies of the base predictors, and $Av_acc_1, ..., Av_acc_n$ are the average accuracies of the LRs.

In general the differences in accuracies among these three fusion methods are not significant. Adding the normalised accuracy to one or two layers can have a positive or a negative impact on the accuracy of the WMV depending on the data sets. In correlation based MCMLPS, the three WMV methods showed comparable accuracies, in few cases the differences in their accuracy exceed 2%. An example is the heart data set, shown in Table 5.13, where similarity based WMV had the highest accuracy for all the cases except when 20% noise is added to the testing data.
In small data sets such as the ionosphere, heart and WBC, using the accuracy in the weighting of the predictive system can lead to a slight decrease in the overall performance of the system. This is due to the possible overfitting of the base predictors when they are trained using small data sets, which can make the training accuracy a poor representation of the base predictors performance. In these cases, using the similarity metric alone can result in a better performance than adding the normalized accuracy to the weighting vector.
On the other hand, compared to other fusion methods, in correlation based MCMLPS the WMV methods have the highest accuracies for most of the data sets. The exceptions

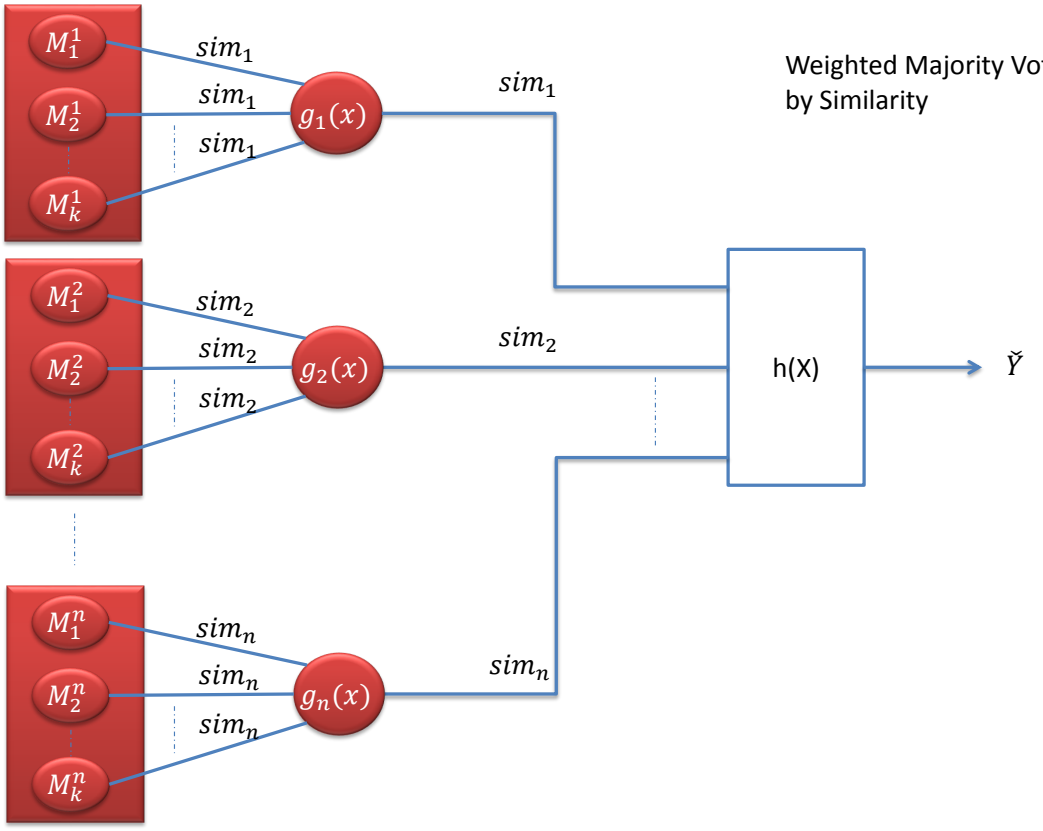


Figure 5.15: *An illustration for the MCMLPS with MV weighted by similarity in all layers*

for this (as was mentioned previously) are: the ionosphere, Pima and sonar data sets. In the ionosphere data set, WMV methods come after the MV fusion method and in the other two data sets it comes after the best model and the MV. Nevertheless, for these data sets the significance of the difference in accuracy between the WMV methods and the best fusion method is reduced when the noise is added to either the testing or the training data.

In MI based MCMLPS, adding the accuracy in the weighing of one or two layers often result in a small improvement in the accuracy of the WMV fusion methods. In this type of MCMLPS, training the base predictors on subsets of the features for all the available data makes them less subjected to overfitting than in the correlation based approach. Due to this, including the accuracy in the weighting of the base predictors/ensembles can adds useful information about the predictors and results in improving the accuracy of the overall system.

Similar to correlation based MCMLPS, in MI based system WMV fusion methods

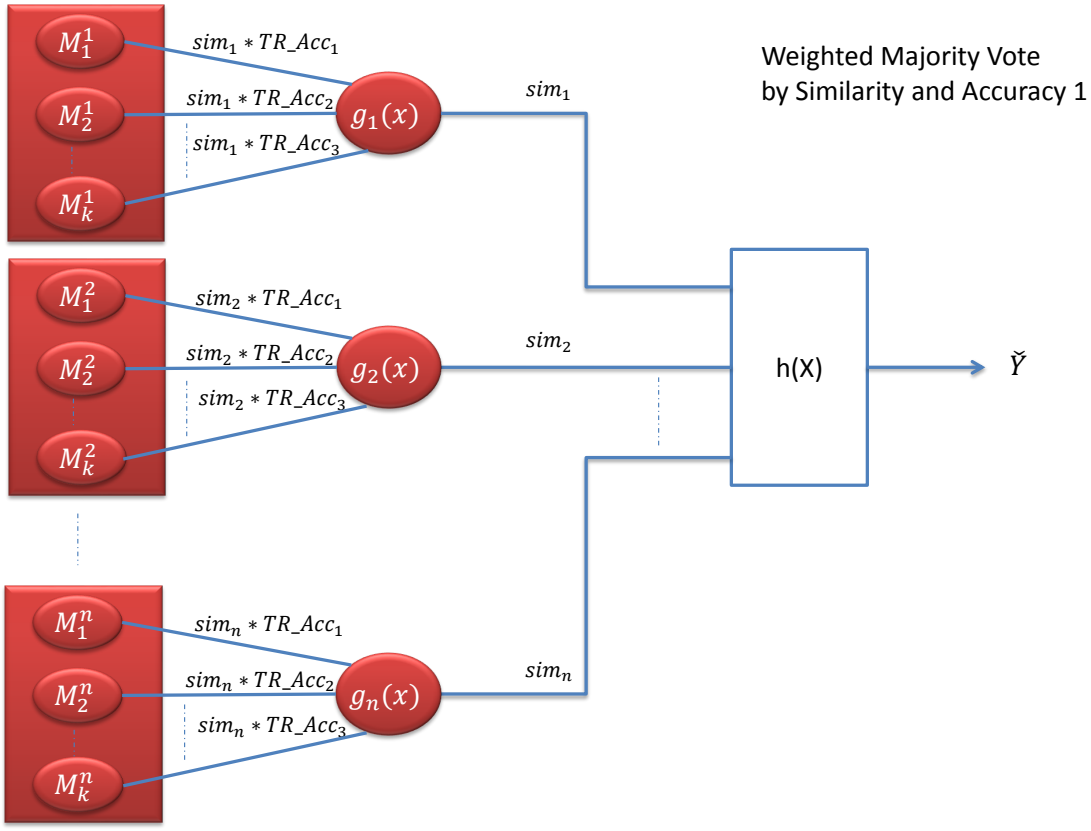


Figure 5.16: *An illustration for the MCMLPS with MV weighted by similarity and normalized training accuracy in the first layer and the similarity in subsequent layers*

outperform other fusion methods in most cases, except when it is outperformed by MV (in the Vehicle data set shown in Table 5.12) or by the single LR (in the ionosphere and German data set shown in Table 5.8 and Table 5.14 respectively). However, the differences in accuracy among these methods are very small.

The standard deviations of the accuracies for all WMV fusion methods using both correlation based and MI based MCMLPS are shown in Table 5.6. It can be noted that within the same MCMLPS, the differences among the three WMV methods are small and that the smallest deviation in accuracy (for correlation based and MI based MCMLPS) depends on the data set used. However, in correlation based MCMLPS, for certain data sets the deviation in accuracy is considerably high. This is due to the high drop in accuracy when the noise is added to the testing data. Nevertheless, this large variation is reduced when MI based MCMLPS is used.

A one way ANOVA test is applied to the three fusion methods of the correlation based MCMLPS: WMV1 ($M = 4.92$), WMV2($M = 4.71$), WMV3($M = 4.72$) and of the MI

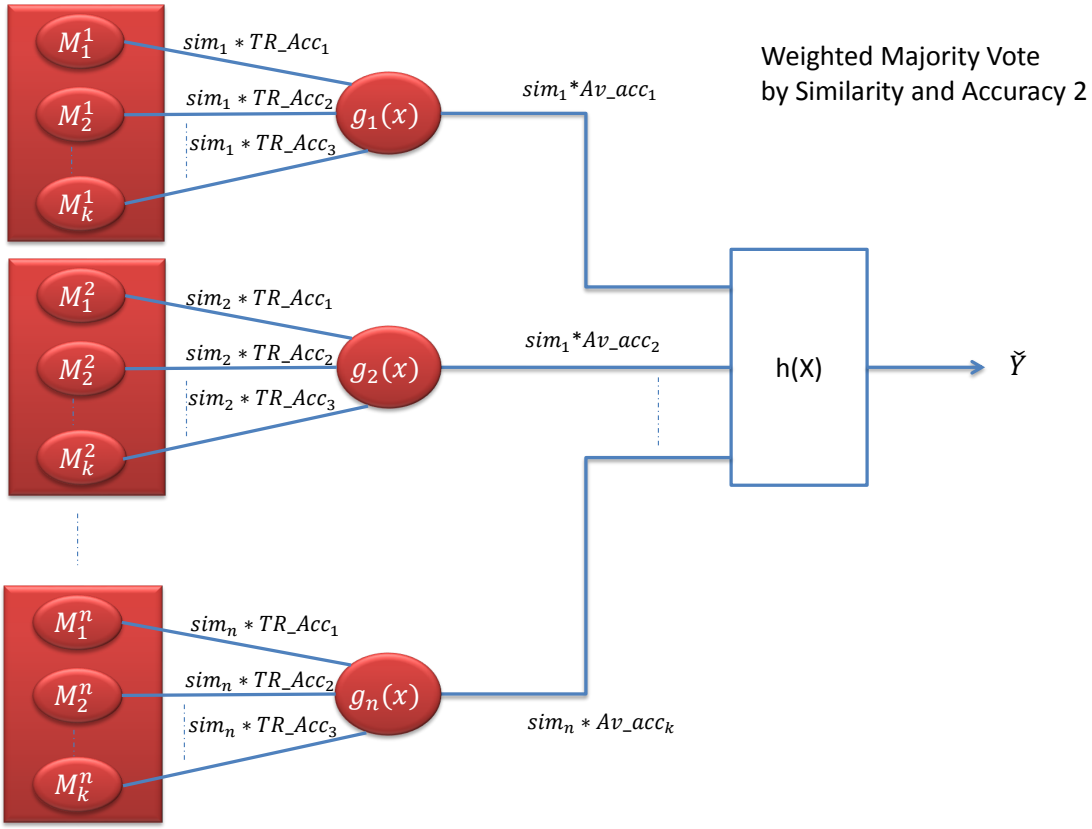


Figure 5.17: *An illustration for the MCMLPS with MV weighted by similarity and normalized training accuracy in the first layer and the similarity and average training accuracy in subsequent layers*

based MCMLPS: WMV1($M = 3.09$), WMV2($M = 3.07$), WMV3($M = 3.08$). The test showed that, though the mean of the MI based MCMLPS is generally lower than that of the correlation based MCMLPS, there is no significant difference between the means of the WMV fusion methods for both systems ($F(5, 60) = 0.65$, $p = 0.66$).

Table 5.6: *The standard deviations of the accuracy for the WMV fusion methods when applied with correlation based and MI based MCMLPS.*

Data sets	Correlation based MCMLPS			MI based MCMLPS		
	WMV 1	WMV 2	WMV 3	WMV 1	WMV 2	WMV 3
Gaussian 8D	10.33	10.38	10.37	3.31	3.24	3.23
German credit	0	0	0	1.95	1.92	1.98
Ionosphere	2.59	1.81	1.94	3.94	3.92	3.92
Spam	9.12	9.30	9.32	3.35	3.31	3.31
Pima	1.94	1.96	1.79	1.84	1.80	1.80
WBC	4.66	4.43	4.02	3.43	3.42	3.45
Heart	3.29	2.00	2.40	2.08	1.99	1.99
Sonar	0.78	0.45	0.44	2.22	2.33	2.33
Chess	3.51	3.54	3.56	3.79	3.79	3.79
Vehicle	17.33	17.33	17.41	4.08	4.10	4.10
waveform	0.54	0.63	0.61	3.95	3.94	3.94

Table 5.7: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the Gaussian 8 dimensions data set.*

Correlation based MCMLPS applied to Gaussian 8 dimensions data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	87.08	87.42	77.86	88.16	88.16	88.16
10%TR	86.50	86.34	74.44	86.82	86.88	86.78
20%TR	86.34	86.32	74.12	86.66	86.60	86.60
10%TS	76.26	75.26	66.04	75.66	75.64	75.66
20%TS	66.14	64.54	57.44	64.16	64.04	64.02

Table 5.8: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the ionosphere data set.*

MI based MCMLPS applied to Ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	92.87	92.87	91.44	92.30	92.58	92.58
10%TR	94.30	94.01	93.44	93.72	94.01	94.01
20%TR	92.87	93.16	93.44	93.72	93.72	93.72
10%TS	90.17	90.17	88.61	89.65	89.91	89.91
20%TS	84.58	85.06	83.63	84.35	84.58	84.58

Table 5.9: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the ionosphere data set.*

Correlation based MCMLPS applied to ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.93	78.35	82.92	77.19	76.06	76.92
10%TR	71.23	76.35	81.19	75.77	74.92	74.63
20%TR	68.96	76.92	74.06	72.37	72.93	73.22
10%TS	71.80	79.49	80.63	77.78	76.92	77.20
20%TS	68.96	76.92	74.06	72.37	72.93	73.22

Table 5.10: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the Pima data set.*

MI based MCMLPS applied to Pima data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	75.13	67.97	71.88	75.78	75.91	75.91
10%TR	71.88	72.14	71.35	75.26	75.39	75.39
20%TR	75.78	73.70	70.83	76.17	76.17	76.17
10%TS	72.27	65.52	69.19	72.87	73.10	73.10
20%TS	71.63	64.35	67.61	72.07	72.17	72.17

Table 5.11: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the sonar data set.*

Correlation based MCMLPS applied to sonar data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.60	78.85	78.85	77.88	77.88	77.88
10%TR	73.56	77.40	78.85	78.37	78.37	78.37
20%TR	75.48	77.40	78.85	78.85	78.85	78.37
10%TS	72.81	77.63	80.26	79.39	78.51	78.07
20%TS	71.77	77.42	79.44	79.84	79.03	79.03

Table 5.12: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the vehicle data set.*

MI based MCMLPS applied to vehicle data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.58	72.46	75.89	74.35	74.59	74.59
10%TR	73.76	74.47	75.06	75.30	75.42	75.42
20%TR	72.69	73.87	76.24	74.82	74.94	74.94
10%TS	68.92	68.60	72.26	70.43	70.65	70.65
20%TS	64.59	64.00	67.46	65.68	65.78	65.78

Table 5.13: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the heart data set.*

Correlation based MCMLPS applied to heart data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.96	75.94	74.84	80.38	78.16	77.03
10%TR	72.55	78.15	77.02	81.49	80.76	80.03
20%TR	73.33	77.79	78.87	81.12	79.27	79.26
10%TS	71.50	74.84	73.50	78.54	77.86	76.17
20%TS	68.25	73.22	69.81	73.47	75.33	74.09

Table 5.14: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the German data set.*

MI based MCMLPS applied to German data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	75	72	69	74.60	74.60	74.60
10%TR	75.10	72.50	69.10	75.80	75.80	76
20%TR	71.70	73.00	67.20	73.90	73.80	73.80
10%TS	73.09	70.27	67.90	72.72	72.72	72.73
20%TS	71.50	68.25	66.17	70.67	70.75	70.75

5.5.5 The relative loss of accuracy for the six fusion methods

The complete results for the RLA for both correlation based MCMLPS and MI based MCMLPS are given in Appendix B (Tables B.1 to B.11 and Tables B.12 to B.22 respectively). In general, the differences between the fusion method that has the highest

RLA and the fusion method that has the lowest RLA are small.
As has been shown in Table 5.2, in MI based MCMLPS, WMV fusion methods often had a lower RLA compared to Bagging, AdaBoost and RF. Nevertheless, compared to the other MCMLPS fusion methods discussed in this Section, WMV methods seldom have the lowest RLA value. Mainly, best model and MV fusion methods have the lowest RLA value in MI based MCMLPS. On the other hand, in correlation based MCMLPS, the fusion methods that have the lowest RLA for most of the data sets are single LR and best model fusion methods.
Table 5.15 shows the number of times each fusion method had the lowest RLA value. This count is measured for the four noise ratios added to the 11 data sets used in this experiment. In correlation based MCMLPS, the results for the German data set, has been excluded, as all the fusion method (apart from MV) had the same RLA value.

Table 5.15: *The count of the lowest RLA values for the six fusion methods*

Fusion method	Correlation based MCMLPS	MI based MCMLPS
Single LR	$11/40$	$6/44$
Best model	$12/40$	$18/44$
MV	$3/40$	$18/44$
WMV with similarity	$4/40$	$3/44$
WMV with similarity and accuracy in $1st$ layer	$7/40$	$0/44$
WMV with similarity and accuracy in all layers	$3/40$	$1/44$

Though including the weighting vectors in WMV methods resulted in improving the accuracy of the system, they had slightly increased the RLA in the overall system. A possible reason for this increase is the increase of the number of free parameters in the system.

5.5.6 Discussion

The previous subsections examined the accuracy and robustness of six fusion methods used to combine the base predictors of the proposed MCMLPS. The robustness is mea-

sured in terms of the standard deviation of the accuracy and the RLA when different noise ratios are added to the training and testing data.

In the correlation based MCMLPS, the LRs ensembles are trained on disjoint sets of the data and for which a subset of features are chosen. Due to this, these ensembles are independent from each other. Choosing a single LR to provide the prediction of the system often resulted in the worse accuracy among the six fusion methods. Though the similarity metric is taken into consideration when choosing the single LR, nevertheless, the base predictors are trained on a limited amount of data and features. This had resulted in having the lowest accuracy among other fusion methods. Furthermore, this fusion method did not benefit from the high diversity among the LRs ensembles (as was discussed in the Chapter 4), since only one of the LRs is chosen to provide the prediction for the system. On the other hand, selecting the best model from each LR and combining them into a single ensemble did improve the accuracy compared to choosing a single LR. In most cases this method had a higher accuracy than MV method; this showed the positive effect of the weighting on the accuracy of the system. Meanwhile, best model fusion method had a lower accuracy than WMV methods, which indicates that combining the base predictors of the LRs should not be considered as an unnecessary step of combining redundant predictors. As has been explained in Chapter 4, the LRs data is split into k-fold using DPS. Though the folds generated using DPS are representative of the LRs data, they are not identical. Thus training the base predictors using these folds does not result in identical predictors.

In general, the WMV had the highest accuracy among the fusion methods for most of the data sets. The differences in the accuracies among the three WMV methods is small. In small data sets including the training accuracy in weighting of the base predictors might lead to a slight decrease in the overall accuracy of the correlation based MCMLPS prediction. This is due to training the base predictors on small disjoint subsets of the data which can result in overfitting.

In MI based MCMLPS, the accuracy of the six fusion methods is improved compared to the correlation based system, since the base predictors are trained on subset of the features for all the available data. In this case, the MV often has the lowest accuracy compared to the other fusion methods, while the WMV methods often have the highest accuracy.

The robustness of the system is investigated by exploring the changes in accuracy when different ratios of noise are added to either the training or the testing data. In MV and

WMV methods, using conditional mutual information to generate the LRs data did not only improve the accuracy of the system but also improved the standard deviation of its accuracy. Moreover, in correlation based MCMLPS, both single LR and best model fusion methods had high values for the standard deviation of the accuracies when applied the Gaussian 8 dimension, spam and vehicle data sets. Using the MI similarity metric had resulted in reducing the standard deviation of the accuracies for those data sets.
On the other hand, measuring the RLA for the six fusion methods showed that, though WMV methods have the highest accuracy values, they often do not have the lowest RLA among the fusion methods. Adding the weights can improve the accuracy of the fusion method, however, it may cause an increase the RLA.

5.6 Summary

This Chapter investigated the performance of the MCMLPS proposed in Chapter 4 in noisy environments. Mainly this Chapter examined the accuracy and the robustness of the MCMLPS generated using correlation based and MI based approaches. The accuracy is evaluated on the test set, while the robustness is evaluated using the RLA and the standard deviation of the accuracies of the system over five noise ratios added to the training and testing data (as was explained in Section 5.4).
The results showed that, there are three main decision points in the proposed MCMLPS that control the accuracy, diversity and robustness of the system. These points are: a) the generation of the LRs (using disjoint sets of the data or subsets of the features), b) the fusion methods (selecting parts of the system components or combining all of them), and c) the weighting of the system components.
In the first part of the experimental work presented in Section 5.4, the accuracy and robustness of the proposed system are compared to the accuracy and robustness of the three ensemble methods, these methods are: Bagging, AdaBoost and RF. MI based MCMLPS often had a comparable accuracy to the best performing method (Bagging). Furthermore, the standard deviation of its accuracy is comparable to Bagging and AdaBoost. Nevertheless, in terms of the RLA when different noise ratios are added to the training and testing data, generally, MI based MCMLPS had a lower loss of accuracy than Bagging.
On the other hand, the correlation based MCMLPS often had a lower accuracy than MI

based MCMLPS, especially for small data sets. Furthermore, the standard deviations of its accuracies vary across the data sets used in this experiment and had reached high values compared to the other ensemble methods.

The second part of the experimental work presented in Section 5.5 introduced six fusion methods to combine the base predictors/ensembles of the MCMLPS. In order to generate the final prediction of the MCMLPS, these fusion methods either combine parts of the system components, such as: single LR and best model fusion methods, or combine all the system components using either a weighted or unweighted MV method.

In terms of the accuracy, usually WMV methods provide the highest accuracy among all six fusion methods. However, in terms of the RLA, WMV methods often have higher loss in accuracy compared to other fusion methods.

Chapter 6

Conclusions and Future Work

6.1 Thesis summary

The goal of this thesis is to investigate the inclusion of multiple criteria in the design process of complex predictive systems and to critically evaluate the interactions among them. The criteria that have been included in this work are: accuracy, model complexity, algorithmic complexity, diversity and robustness.

This thesis started with decomposing the learning process in predictive systems into three components: representation, evaluation and optimisation. The literature reviewed in Chapter 2 explored these components in order to establish the theoretical background for the work presented in this thesis. It started by defining the architectures of the predictive systems, from single predictor to complex pool of competing predictors. Then, this Chapter had examined the criteria used to evaluate the predictive system performance and examined the main approaches used to optimise a single as well as multiple criteria.

Taking the three components of the learning process into consideration, a general design cycle for building MCMLPS has been proposed in Chapter 3. This design cycle had considered important aspects of MCMLPS, such as: local versus global models, the ability of defining universal measures for the evaluation criteria and the optimisation approaches used for such systems.

Chapter 3 introduced an experimental case study to compare the models generated from trading-off accuracy, model complexity and algorithmic complexity using two MOO approaches. This case study evaluated the significance of including multiple criteria at the base predictor's level of the MCMLPS design process. Furthermore, it highlighted the

high algorithmic complexity associated with including multiple criteria in the optimisation of the base models.

In addition, Chapter 3 had illustrated the advantages and drawbacks of using scalarized and Pareto-based MOO approaches in optimising multiple criteria experimentally using the above mentioned case study. The results showed that, the best models generated from the Pareto-based approach usually had lower error than the models generated from the scalarized approach. However, the Pareto-based approach is hindered by its high algorithmic complexity. Thus, in ensemble methods, the performance of the base predictors is often evaluated using only the accuracies of the models prediction. While other characteristics of the system are either included indirectly in the design cycle or presented as constraints.

In Chapter 4, a novel locally trained MCMLPS has been proposed. This system divided the data into multiple LRs using the similarity of the features. Inside each LR a pre-defined number of base models were trained on subsets of data and/or subsets of features. Two similarity metrics were used: pairwise squared correlation and conditional mutual information. The squared correlation metric can be applied in supervised as well as unsupervised learning since it does not require the output class when splitting the data. Meanwhile, the conditional mutual information metric can be applied only in supervised learning as it uses the output class while splitting the data.

The diversity among the MCMLPS base predictors was relatively high compared to the other ensemble methods described in Chapter 4, especially when the base predictors were trained on disjoint subsets of the data. The locality and the high level of diversity among the base predictors of the MCMLPS suggested that this system can have more robust performance than the benchmark ensembles when applied in noisy environments. Thus, the robustness of the proposed system to external noise had been investigated in the next Chapter.

Chapter 5 examined the robustness of the proposed MCMLPS and its relation to the locality of the data, the diversity of the base predictors and the accuracy of the prediction. The robustness was evaluated using both the RLA and the standard deviation of the accuracies over five noise ratios added to either the training or the testing data. Though the accuracy of the MI based MCMCPS was comparable to the best performing method (when no noise is added to the data), its RLA (when different noise ratios are added to the data) is often lower than the best performing method.

The effect of changing both the weighting and the fusion methods on the accuracy and

robustness of the MCMLPS was examined in this Chapter. Six fusion methods were introduced to combine the base models/ensembles of the proposed system. In order to generate the final prediction of the MCMLPS, these fusion methods had either combined parts of the system components, such as: single LR and best model fusion methods, or combined all of the system components using either weighted or unweighted MV methods.

In terms of the accuracy, usually WMV methods had the highest accuracy among all six fusion methods. However, in terms of the RLA, WMV methods often had a higher loss in accuracy compared to other fusion methods.

Chapter 5 concluded by identifying the main decision points in the system that control its accuracy, diversity and robustness to noise. The identified points were: a) the generation of the LRs (using disjoint sets of the data or subsets of the features), b) the fusion methods (selecting parts of the system components or combining all of them), and c) the weighting of the system components.

6.2 Main contributions

The main contributions of this work can be summarised in the following points:

- Comprehensive study of the learning process in predictive systems:
  The study followed the decomposition of the learning process presented in (Domingos (2012)). It carried out a survey of the prediction systems architectures, evaluation criteria and optimisation techniques. The scope of this theoretical study was to identify the major criteria used in evaluating the system performance and to determine the feasibility of defining universal measurements for the considered criteria. Moreover, it explained and compared the optimisation approaches used to optimise single as well as multiple criteria.
- An experimental case study for multi-criteria optimisation:
  This study compared the scalarized and Pareto-based MOO approaches employed to generate predictive models using multiple criteria. The performance of the models was optimised using their accuracy, model complexity and algorithmic complexity (captured through the execution time and the memory usage). The results showed that Pareto-based optimisation approaches can produce more accurate and more diverse models.

- Novel local learning MCMLPS:
  Two versions of novel local learning MCMLPS were developed. In both versions, the diversity among the system base predictors was maximised by training their base predictors on subsets of the data or the features.
  - Correlation based MCMLPS:
    In this system the data was split into disjoint subsets that were assigned to a set of LRs. The locality was determined using an unsupervised similarity metric, that is the pairwise squared correlation of the features. A pre-defined number of models were trained on the LRs data and their output were combined using WMV. The similarity between the LRs seeds and the features of the data were used to weight the prediction of the system components. A particular benefit of this MCMLPS is that, since it had trained the LRs on disjoint subsets of the data, the diversity among its components were maximised.
  - MI based MCMLPS:
    This system had trained its base models on intersected subsets of the features for all the training data. The locality of the features was determined using a supervised similarity metric, namely, the conditional mutual information of the features. The weighting and the fusion method used in this approach were similar to the previous system.
    The diversity among the components of MI based MCMLPS was lower than that of the correlation based MCMLPS. Nevertheless, it was higher than the diversity among the components of the other ensemble methods used in the experiments. Furthermore, the accuracy of the prediction for this system had improved compare to that of the correlation based MCMLPS.
- Robustness of MCMLPS in noisy environments:
  Encouraged by the high diversity among the MCMLPS base models, the robustness of this system in noisy environments was tested and compared to other ensemble methods. The results showed that, the MI based MCMLPS had: a) an accuracy comparable to the best performing method, b) a lower diversity than correlation based MCMLPS and c) often lower RLA than the best performing method. Thus, compared to the ensemble with the highest accuracy, the performance of this approach was often less effected by the noise added to the data.

- Examining the effect of changing the fusion methods on the performance of the MCMLPS:

  The effect of changing the fusion/selection methods and the weighting of the prediction on the robustness and accuracy of the prediction had been examined. Six fusion methods that use different weighting vectors were applied to both correlation based and MI based MCMLPS. Comparing the fusion methods showed that, combining the base predictors using WMV often had the best accuracy, though its RLA can be slightly higher than other combining methods. Furthermore, the benefits of using the similarity metric in weighting the prediction of the system was highlighted when the results were compared to unweighted MV fusion method which often had a lower accuracy.

  Generally, there are three main decision points in the proposed MCMLPS that control the accuracy, diversity and robustness of the system, these points are:

  - Data partitioning: which defines how the data is split and used to train the base predictors.
  - Prediction weighting: which determines how the predictions of the base predictors/ensembles are weighted.
  - Fusion methods: which defines how to combine the base predictors/ensembles.

6.3 Future work

There are at least two possible directions for the future research related to the work presented in this thesis. The first direction is related to the development and expansion of the current architecture of the system, while, the second direction is related to the type of applications this system can be particularly useful in.

In terms of the first direction, there are certain aspects in the current setting of the system that can be further improved, such as, adding a subroutine to optimise the number of LRs and number of models inside the LRs to match the data set size, dimensionality and number of classes. The current setting of the experimental work in this thesis had pre-defined these parameters to match the overall number of models in the benchmark ensemble methods used in the comparison.

In addition, the functionality of the proposed MCMLPS can be extended to include

adaptation mechanisms, such that, when a change is detected in the data, only the models of the affected region are updated. This can reduce the algorithmic complexity required to update the system. Another extension is to develop a meta-learning layer that can select the type of the base models according to the nature of the prediction problem rather than having a predefined type of base models. This extension can potentially improve the accuracy of the system. This was shown in the results of the accuracy for correlation based MCMLPS (in Chapter 4), where changing the type of the base predictors from CART decision trees to feedforward NNs had improved the accuracy of the prediction for certain data sets. In this case, it would be beneficial to find other approaches to introduce multiple criteria (such as model and algorithmic complexities) at the generation level of the base components without increasing much the algorithmic complexity of the system.

In terms of the area of applications, due to the locality and high diversity of the proposed MCMLPS, it can be particularly useful when applied to data from different sources (fused data) (Parikh and Polikar (2007)). On the other hand, the correlation based similarity metric used in this work has been applied successfully with deep learning networks in challenging image recognition problems (Coates and Ng (2011)). Since the proposed system is trained to recognize local regions in the data, it could be potentially successful when applied to image recognition problems. Furthermore, the high diversity among the system components combined with low RLA (in the MI based MCMLPS) could have a beneficial effect in recognizing images with distortion or added noise. In addition, in this thesis, the proposed MCMLPS has been applied to classification problems, however, the current structure of the system can be extended to provide prediction for regression problems as well.

Appendix A

Data Sets Descriptions

This appendix provides a summary for the data sets that have been used throughout the thesis. The data sets are publicly available and come from the UCI machine learning repository Lichman (2013) and the UCL ELENA project. The data sets have been chosen according to their size, dimensionality and number of classes. Descriptions of these data sets are given below:

- Cloud data set
  The cloud data set is an artificial data set available at the ELENA project database. It has $2-$dimension and consist of 5000 instances. This data set represent a heavily intersected binary classification problem, where class 0 is obtained from summing three Gaussian distributions, while class 1 is a single normal distribution.
- Concentric data set
  The concentric data set is also an artificial data set available at the ELENA project database. It has $2-$dimensions and consist of 2500 instances. This data set represent a uniform concentric circular distributions with two classes. The dataset is contained in the square (0,0), (1,1), where class 0 is uniformly distributed into a circle of radius 0.3 centred on (0.5,0.5). On the hand, class 1 is uniformly distributed into a ring centred on (0.5,0.5) with internal and external radius respectively equal to 0.3 and 0.5.
- Cone-Torus data set
  Conetorus is a synthetic data set dataset first used in Kuncheva (2000). It has 2 dimensions and consisting of 3 classes. The classes are generated from 3 differently shaped distributions: a cone, half a torus, and a normal distribution with prior

probabilities of 0.25, 0.25 and 0.5 respectively.
The dataset comes divided into nonoverlapping training and test sets, each has 400 instances.

- Gaussian 2,4 and 8 dimensional data sets
  The Gaussian database is a group of artificial datasets available at the ELENA project database. They represent binary classification problems and has 5000 instances. The number of features for this data sets ranges from 2 to 8 dimensions. This problem explore the classifier behaviour for different dimensionalities of the same data, under heavily overlapped distributions with no linear separability.
  The first class of this data represents multivariate normal distribution with zero mean and unit standard deviation. On the other hand the second class represents normal distribution with zero mean and standard deviation equal to 2 in all dimensions.
- Shuttle data set
  The shuttle dataset (available at the UCI repository) is concerns automatic shuttle control. It has 9 numerical features and 7 classes. These classes correspond to control actions, with 80% of instances belonging to one majority class. This dataset consists of 58000 instances.
- Synthetic data set
  Synthetic dataset is an artificial first used in Ripley (2007). It has 2 dimension and consist of 2 classes with equal class priors. The two classes are partially overlapping and they are composed of 2 Gaussians distributions with shifted centres.
  The dataset comes divided into nonoverlapping training and test sets, which have 250 and 1000 instances respectively.
- German credit card data set
  The German credit card data set (available at the UCI repository) is concerned with identifying good and bad credit risk for loan applicants. It has 24 features and consist of 1000 instances. It represent a cost sensitive prediction problem, where the cost associated with identifying bad loan application as good is higher than identifying good loan application as bad.
- Ionosphere data set
  The ionosphere radar data (available at the UCI repository) was collected by a system in Goose Bay. It has 34 features and 351 instances. This is a binary clas-

sification problem, where the first class represent Good radar returns for electrons that shows evidence of some type of structure in the ionosphere. On the other hand the second class represent Bad returns for electrons that do not show such structure.

- Spam base data set
  The spam base data set (available at the UCI repository) is concerned with identifying emails as spam or non-spam, thus it is a binary classification problem. It has 57 features and consist of 4601 instances.
- Pima Indian diabetes
  Pima Indians diabetes database (available at the UCI repository) providing diabetic diagnosis based on a number of various physiological attributes of the examined patients. All patients are females who are at least 21 years old of Pima Indian heritage. This data set has 8-features and consist of 768 instances. It is a binary classification problem, where 500 instances belong to class 1 while 268 belong to class 2.
- Wisconsin Breast Cancer data set
  The breast cancer database (available at the UCI repository) is a data set that has been obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg. It is a binary classification problem that has 30 features and consist of 596 instances. The two classes of this data are benign and malignant breast cancers and they are split into 212 malignant instances and 357 benign instances. The features of the data were computed from a digitised image of a fine needle aspirate of a breast mass.
- Heart data set
  The breast cancer database (available at the UCI repository) is a binary classification problem that has 13 features and consist of 270 instances. The two classes of this data are absence and presence of heart disease.
- Sonar data set
  The sonar dataset (available at the UCI repository) has 60 features and consist of 208 instances. It is a binary classification problem which distinguish between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock.
- Chess
  Chess (King-Rook vs. King-Pawn) dataset is available at the UCI repository. It

has been originally generated and described by Alen Shapiro. This data set has 36 features and consist of 3196 instances. It is a binary classification problem for a win or no win scenario in chess board. The class distribution consist of 1669 of the positions (52%) where white can win and 1527 of the positions (48%) where white cannot win.

- Vehicle

  Vehicle silhouettes dataset (available at the UCI repository) has 18 features and consist of 846 instances. This data set distinguish four types of vehicles using their silhouettes. These vehicles are Opel, Saab, Bus and Van. The vehicle may be viewed different angles.

- Waveform

  Waveform dataset (available at the UCI repository) has 40 features all of which include noise and consist of 5000 instances. This data set distinguish three equally distributed classes. Each class is generated from a combination of 2 of 3 ”base” waves.

Appendix B

The accuracy and RLA for the MCMLPS fusion methods

This appendix provide the detailed results for the accuracy and the RLA for the MCMLPS fusion methods. Chapter 5 provide a selective set of results that shows certain aspect of the fusion methods. In this section the results for the accuracy and the RLA for the six fusion methods when applied to the data sets shown in Table 4.3 are given below:

B.1 The accuracy of the six fusion methods

In this section the accuracy of the six fusion methods when applied to the correlation based and MI based MCMLPS are given in Tables B.1 to B.11 and Tables B.12 to B.22 respectively.

B.1.1 Accuracy for the six fusion methods in the correlation based MCMLPS

- Gaussian 8 dimensional data set

Table B.1: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the Gaussian 8 dimensions data set.*

Correlation based MCMLPS applied to Gaussian 8 dimensions data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	87.08	87.42	77.86	88.16	88.16	88.16
10%TR	86.50	86.34	74.44	86.82	86.88	86.78
20%TR	86.34	86.32	74.12	86.66	86.60	86.60
10%TS	76.26	75.26	66.04	75.66	75.64	75.66
20%TS	66.14	64.54	57.44	64.16	64.04	64.02

- German credit card data set

Table B.2: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the German credit card data set.*

Correlation based MCMLPS applied to German credit data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	70	70	69.80	70	70	70
10%TR	70	70	69.70	70	70	70
20%TR	70	70	69.50	70	70	70
10%TS	70	70	66.60	70	70	70
20%TS	70	70	68.30	70	70	70

- Ionosphere data set

Table B.3: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the ionosphere data set.*

Correlation based MCMLPS applied to ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.93	78.35	82.92	77.19	76.06	76.92
10%TR	71.23	76.35	81.19	75.77	74.92	74.63
20%TR	68.96	76.92	74.06	72.37	72.93	73.22
10%TS	71.80	79.49	80.63	77.78	76.92	77.20
20%TS	68.96	76.92	74.06	72.37	72.93	73.22

- Spam base data set

Table B.4: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the spam base data set.*

Correlation based MCMLPS applied to spam base data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	82.61	82.24	85.98	86.26	86.29	86.29
10%TR	80.48	84.44	84.92	83.37	83.53	83.53
20%TR	81.09	84.07	84.33	84.18	84.74	84.74
10%TS	65.60	64.06	66.29	68.15	68.03	67.95
20%TS	66.06	63.90	62.94	67.98	67.89	67.91

- Pima Indian diabetes

Table B.5: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the Pima data set.*

Correlation based MCMLPS applied to Pima data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.08	78.06	81.50	76.62	75.20	76.35
10%TR	70.95	78.34	81.47	75.77	75.78	75.21
20%TR	71.23	75.77	77.50	75.50	75.21	75.22
10%TS	71.52	76.63	80.63	77.50	77.50	77.50
20%TS	68.96	75.78	74.35	72.37	72.08	72.66

- Wisconsin Breast Cancer data set

Table B.6: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the WBC data set.*

Correlation based MCMLPS applied to WBC data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	78.38	82.60	81.20	86.29	85.77	85.06
10%TR	76.97	83.31	80.85	85.06	84.36	83.66
20%TR	80.14	81.54	77.33	84.36	84.53	83.66
10%TS	74.24	78.56	75.68	81.60	81.45	81.45
20%TS	71.25	73.90	72.59	74.64	74.79	74.94

- Heart data set

Table B.7: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the heart data set.*

Correlation based MCMLPS applied to heart data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.96	75.94	74.84	80.38	78.16	77.03
10%TR	72.55	78.15	77.02	81.49	80.76	80.03
20%TR	73.33	77.79	78.87	81.12	79.27	79.26
10%TS	71.50	74.84	73.50	78.54	77.86	76.17
20%TS	68.25	73.22	69.81	73.47	75.33	74.09

- Sonar data set

Table B.8: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the sonar data set.*

Correlation based MCMLPS applied to sonar data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.60	78.85	78.85	77.88	77.88	77.88
10%TR	73.56	77.40	78.85	78.37	78.37	78.37
20%TR	75.48	77.40	78.85	78.85	78.85	78.37
10%TS	72.81	77.63	80.26	79.39	78.51	78.07
20%TS	71.77	77.42	79.44	79.84	79.03	79.03

- Chess

Table B.9: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the chess data set.*

Correlation based MCMLPS applied to chess data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	93.52	94.09	93.18	94.15	94.02	94.06
10%TR	93.02	94.99	93.87	94.62	94.56	94.56
20%TR	93.74	94.96	94.43	94.96	94.96	95.02
10%TS	89.19	89.82	89.65	89.90	89.68	89.73
20%TS	86.50	86.52	86.08	87.02	86.97	86.94

- Vehicle

Table B.10: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the Vehicle data set.*

Correlation based MCMLPS applied to Vehicle data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	58.98	62.41	67.73	67.61	68.44	68.09
10%TR	55.32	62.53	65.13	67.03	67.02	67.50
20%TR	55.44	58.51	64.19	66.79	65.72	65.61
10%TS	31.08	37.94	34.28	36.17	35.46	34.63
20%TS	32.98	32.50	31.56	34.88	35.47	36.06

- Waveform

Table B.11: *The accuracy of the fusion methods for the correlation based MCMLPS when applied to the Waveform data set.*

Correlation based MCMLPS applied to Wavefrom data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	26.32	56.62	58.96	65.50	65.60	65.68
10%TR	25.94	55.40	58.38	65.90	65.90	64.88
20%TR	25.18	54.26	56.04	64.64	64.44	66.40
10%TS	25.25	58.02	59.05	65.98	65.91	66.32
20%TS	25.15	58.47	59.23	65.27	65.07	65.68

B.1.2 The accuracy for the six fusion methods in the MI based MCMLPS

- Gaussian 8 dimensional data set

Table B.12: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the Gaussian 8 dimensional data set.*

MI based MCMLPS applied to Gaussian 8 dimensions data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	86.88	85.74	86	86.94	87.18	87.14
10%TR	86.42	86.28	86.66	87.98	87.94	87.90
20%TR	86.46	86.44	86.68	88.48	88.50	88.44
10%TS	83.56	82.44	82.84	83.60	83.76	83.75
20%TS	80.75	79.68	79.97	80.70	80.88	80.85

- German credit card data set

Table B.13: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the German data set.*

MI based MCMLPS applied to German data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	75	72	69	74.60	74.60	74.60
10%TR	75.10	72.50	69.10	75.80	75.80	76
20%TR	71.70	73.00	67.20	73.90	73.80	73.80
10%TS	73.09	70.27	67.90	72.72	72.72	72.73
20%TS	71.50	68.25	66.17	70.67	70.75	70.75

- Ionosphere data set

Table B.14: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the ionosphere data set.*

MI based MCMLPS applied to Ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	92.87	92.87	91.44	92.30	92.58	92.58
10%TR	94.30	94.01	93.44	93.72	94.01	94.01
20%TR	92.87	93.16	93.44	93.72	93.72	93.72
10%TS	90.17	90.17	88.61	89.65	89.91	89.91
20%TS	84.58	85.06	83.63	84.35	84.58	84.58

- Spam base data set

Table B.15: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the spam base data set.*

MI based MCMLPS applied to spam base data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	93.57	93.50	91.46	93.81	93.89	93.89
10%TR	93.46	93.48	91.26	93.68	93.76	93.76
20%TR	93.98	93.46	91.55	94.15	94.00	94.00
10%TS	89.78	89.69	87.85	89.96	90.04	90.04
20%TS	86.29	86.18	84.57	86.45	86.54	86.54

- Pima Indian diabetes

Table B.16: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the Pima data set.*

MI based MCMLPS applied to Pima data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	75.13	67.97	71.88	75.78	75.91	75.91
10%TR	71.88	72.14	71.35	75.26	75.39	75.39
20%TR	75.78	73.70	70.83	76.17	76.17	76.17
10%TS	72.27	65.52	69.19	72.87	73.10	73.10
20%TS	71.63	64.35	67.61	72.07	72.17	72.17

- Wisconsin Breast Cancer data set

Table B.17: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the WBC data set.*

MI based MCMLPS applied to WBC data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	94.55	93.85	91.21	95.61	95.61	95.61
10%TR	94.90	94.55	92.62	95.61	95.61	95.78
20%TR	94.37	95.60	92.97	95.78	95.78	95.78
10%TS	90.56	89.76	87.36	91.52	91.52	91.52
20%TS	86.95	86.51	84.30	88.11	88.12	88.12

- Heart data set

Table B.18: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the heart data set.*

MI based MCMLPS applied to heart data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	77.79	69.63	70.76	78.89	78.89	78.89
10%TR	78.54	76.31	72.24	80.40	80.40	80.40
20%TR	80.40	76.28	71.12	80.76	80.39	80.39
10%TS	76.86	69.14	69.82	77.86	77.86	77.86
20%TS	74.71	67.00	67.95	75.63	75.63	75.63

- Sonar data set

Table B.19: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the sonar data set.*

MI based MCMLPS applied to sonar data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	80.77	80.29	80.29	84.62	84.13	84.13
10%TR	82.69	78.37	81.25	84.13	84.13	84.13
20%TR	76.92	81.25	83.17	83.17	83.17	83.17
10%TS	76.32	77.63	76.32	80.26	79.82	79.82
20%TS	75.81	76.61	76.61	79.84	79.44	79.44

- Chess data set

Table B.20: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the chess data set.*

MI based MCMLPS applied to chess data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	97.93	98.44	97.87	98.78	98.78	98.78
10%TR	98.09	98.75	97.97	98.81	98.81	98.81
20%TR	98.22	98.81	98.15	99.03	99.03	99.03
10%TS	93.60	94.06	93.57	94.40	94.40	94.40
20%TS	89.75	90.25	89.78	90.48	90.48	90.48

- Vehicle

Table B.21: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the vehicle data set.*

MI based MCMLPS applied to vehicle data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	72.58	72.46	75.89	74.35	74.59	74.59
10%TR	73.76	74.47	75.06	75.30	75.42	75.42
20%TR	72.69	73.87	76.24	74.82	74.94	74.94
10%TS	68.92	68.60	72.26	70.43	70.65	70.65
20%TS	64.59	64.00	67.46	65.68	65.78	65.78

- Waveform

Table B.22: *The accuracy of the fusion methods for the MI based MCMLPS when applied to the waveform data set.*

MI based MCMLPS applied to waveform data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
0%	80.60	79.60	79.60	81.80	82.02	82.04
10%TR	79.58	80.22	79.22	82.10	82.36	82.40
20%TR	80.30	80.40	80.58	82.56	82.60	82.50
10%TS	76.45	75.42	75.62	77.53	77.67	77.67
20%TS	72.20	71.48	71.43	73.40	73.62	73.60

B.2 The relative loss of accuracy in the six fusion methods

In this section the RLA for the six fusion methods when applied to the correlation based and MI based MCMLPS are given in Tables B.23 to B.33 and Tables B.34 to B.44 respectively

B.2.1 The RLA for the six fusion methods in the correlation based MCMLPS

- Gaussian 8 dimensional data set

Table B.23: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the Gaussian 8 dimensions data set.*

Correlation based MCMLPS applied to Gaussian 8 dimensions data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.007	0.0123	0.0439	0.0151	0.0145	0.0157
20%TR	0.008	0.0126	0.0480	0.0170	0.0177	0.0177
10%TS	0.124	0.1390	0.1518	0.1418	0.1420	0.1418
20%TS	0.240	0.2617	0.2623	0.2722	0.2736	0.2738

- German credit card data set

Table B.24: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the German credit card data set.*

Correlation based MCMLPS applied to German credit card data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0	0	0.0014	0	0	0
20%TR	0	0	0.0043	0	0	0
10%TS	0	0	0.0458	0	0	0
20%TS	0	0	0.0215	0	0	0

- Ionosphere data set

Table B.25: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the ionosphere data set.*

Correlation based MCMLPS applied to ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0233	0.02553	0.0209	0.0184	0.0150	0.0298
20%TR	0.0544	0.01825	0.1068	0.0624	0.0412	0.0481
10%TS	0.0155	-0.0146	0.0276	-0.0076	-0.0113	-0.0036
20%TS	0.0544	0.0183	0.1068	0.06244	0.0411	0.04810

- Spam base data set

Table B.26: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the spam base data set.*

Correlation based MCMLPS applied to spam base data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0258	-0.0268	0.0123	0.0335	0.0320	0.0320
20%TR	0.0184	-0.0223	0.0192	0.0241	0.0180	0.0180
10%TS	0.2059	0.2211	0.2290	0.2099	0.2116	0.2125
20%TS	0.2003	0.2230	0.2680	0.2119	0.2132	0.2130

- Pima Indian diabetes

Table B.27: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the Pima data set.*

Correlation based MCMLPS applied to Pima data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0157	-0.0036	0.0004	0.0111	-0.0077	0.0149
20%TR	0.0118	0.0293	0.0491	0.0146	-0.0001	0.0148
10%TS	0.0078	0.0183	0.0107	-0.0115	-0.0306	-0.0151
20%TS	0.0433	0.0292	0.0877	0.0555	0.041499	0.04833

- Wisconsin Breast Cancer data set

Table B.28: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the WBC data set.*

Correlation based MCMLPS applied to WBC data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0180	-0.0086	0.0043	0.0143	0.0164	0.0165
20%TR	-0.0225	0.0128	0.0477	0.0224	0.0145	0.0165
10%TS	0.0528	0.0489	0.0680	0.0544	0.0504	0.0424
20%TS	0.0910	0.1053	0.1060	0.1350	0.1280	0.1190

- Heart data set

Table B.29: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the heart data set.*

Correlation based MCMLPS applied to heart data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0056	-0.0291	-0.0291	-0.0138	-0.0333	-0.0389
20%TR	-0.0051	-0.0244	-0.0538	-0.0092	-0.0142	-0.0289
10%TS	0.0200	0.0145	0.0179	0.0229	0.0038	0.0112
20%TS	0.0646	0.0358	0.0672	0.0860	0.0362	0.0382

- Sonar data set

Table B.30: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the sonar data set.*

Correlation based MCMLPS applied to sonar data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0132	0.0184	0	-0.0063	-0.0063	-0.0063
20%TR	-0.0397	0.0184	0	-0.0125	-0.0125	-0.0063
10%TS	-0.0029	0.0155	-0.0179	-0.0193	-0.0081	-0.0024
20%TS	0.0114	0.0181	-0.0075	-0.0252	-0.0148	-0.0148

- Chess

Table B.31: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the chess data set.*

Correlation based MCMLPS applied to chess data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0053	-0.0096	-0.0074	-0.0050	-0.0057	-0.0053
20%TR	-0.0024	-0.0092	-0.0134	-0.0086	-0.0100	-0.0102
10%TS	0.0463	0.0454	0.0379	0.0451	0.0462	0.0460
20%TS	0.0751	0.0805	0.0762	0.0757	0.0750	0.0757

- Vehicle

Table B.32: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the vehicle data set.*

Correlation based MCMLPS applied to vehicle data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0621	-0.0019	0.0384	0.0086	0.0207	0.0087
20%TR	0.0600	0.0625	0.0522	0.0121	0.0397	0.0364
10%TS	0.4730	0.3921	0.4939	0.4650	0.4819	0.4914
20%TS	0.4408	0.4793	0.5340	0.4841	0.4817	0.4704

- Waveform

Table B.33: *The RLA of the fusion methods for the correlation based MCMLPS when applied to the waveform data set.*

Correlation based MCMLPS applied to waveform data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0144	0.0215	0.0098	-0.0061	0.0122	0.0124
20%TR	0.0433	0.04168	0.0495	0.0131	-0.0110	0.0238
10%TS	0.0407	-0.0247	-0.0015	-0.0073	-0.0097	-0.0108
20%TS	0.0445	-0.03267	-0.0046	0.0035	0	-0.0049

B.2.2 The RLA for the six fusion methods in the MI based MCMLPS

- Gaussian 8 dimensional data set

Table B.34: *The RLA of the fusion methods for the MI based MCMLPS when applied to the Gaussian 8 dimensional data set.*

MI based MCMLPS applied to Gaussian 8 dimensions data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0053	-0.0063	-0.0077	-0.0120	-0.0087	-0.0087
20%TR	0.0048	-0.0082	-0.0079	-0.0177	-0.0151	-0.0149
10%TS	0.0382	0.0385	0.0367	0.0384	0.0392	0.0389
20%TS	0.0706	0.0707	0.0701	0.0718	0.0723	0.0722

- German credit card data set

Table B.35: *The RLA of the fusion methods for the MI based MCMLPS when applied to the German data set.*

MI based MCMLPS applied to German data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0013	-0.0069	-0.0014	-0.0161	-0.0161	-0.0188
20%TR	0.0440	-0.0139	0.0261	0.0094	0.0107	0.0107
10%TS	0.0255	0.0240	0.0159	0.0252	0.0252	0.0251
20%TS	0.0467	0.0521	0.0410	0.0527	0.0516	0.0516

- Ionosphere data set

Table B.36: *The RLA of the fusion methods for the MI based MCMLPS when applied to the ionosphere data set.*

MI based MCMLPS applied to ionosphere data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0154	-0.0123	-0.0219	-0.0154	-0.0154	-0.0154
20%TR	0	-0.0031	-0.0219	-0.0154	-0.0123	-0.0123
10%TS	0.0291	0.0291	0.0309	0.0287	0.0288	0.0288
20%TS	0.0893	0.0841	0.0854	0.0861	0.0864	0.0864

- Spam base data set

Table B.37: *The RLA of the fusion methods for the MI based MCMLPS when applied to the spam base data set.*

MI based MCMLPS applied to spam base data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0012	0.0002	0.0022	0.0014	0.0014	0.0014
20%TR	-0.0044	0.0004	-0.0010	-0.0036	-0.0012	-0.0012
10%TS	0.0405	0.0407	0.0395	0.0410	0.0410	0.0410
20%TS	0.0778	0.0783	0.0753	0.0785	0.0783	0.0783

- Pima Indian diabetes

Table B.38: *The RLA of the fusion methods for the MI based MCMLPS when applied to the Pima data set.*

MI based MCMLPS applied to Pima data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0433	-0.0614	0.0074	0.0069	0.0069	0.0069
20%TR	-0.0087	-0.0843	0.0146	-0.0051	-0.0034	-0.0034
10%TS	0.0381	0.0360	0.0374	0.0384	0.0370	0.0370
20%TS	0.0466	0.0533	0.0594	0.0490	0.0493	0.0493

- Wisconsin Breast Cancer data set

Table B.39: *The RLA of the fusion methods for the MI based MCMLPS when applied to the WBC data set.*

MI based MCMLPS applied to WBC data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0037	-0.0075	-0.0155	0	0	-0.0018
20%TR	0.0019	-0.0186	-0.0193	-0.0018	-0.0018	-0.0018
10%TS	0.0422	0.0436	0.0422	0.0428	0.04278	0.0428
20%TS	0.0804	0.0782	0.0758	0.0784	0.07834	0.0783

- Heart data set

Table B.40: *The RLA of the fusion methods for the MI based MCMLPS when applied to the heart data set.*

MI based MCMLPS applied to heart data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0096	-0.0959	-0.0209	-0.0191	-0.0191	-0.0191
20%TR	-0.0336	-0.0955	-0.0051	-0.0237	-0.0190	-0.0190
10%TS	0.0120	0.0070	0.0133	0.0131	0.0131	0.0131
20%TS	0.0396	0.0378	0.0397	0.0413	0.0413	0.0413

- Sonar data set

Table B.41: *The RLA of the fusion methods for the MI based MCMLPS when applied to the sonar data set.*

MI based MCMLPS applied to sonar data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0238	0.0239	-0.0120	0.0058	0	0
20%TR	0.0477	-0.0120	-0.0359	0.0171	0.0114	0.0114
10%TS	0.0551	0.0331	0.04945	0.0515	0.0512	0.0512
20%TS	0.0614	0.0458	0.0458	0.0565	0.0557	0.0557

- Chess data set

Table B.42: *The RLA of the fusion methods for the MI based MCMLPS when applied to the chess data set.*

MI based MCMLPS applied to chess data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0016	-0.0031	-0.0010	-0.0003	-0.0003	-0.0003
20%TR	-0.0030	-0.0038	-0.0029	-0.0025	-0.0025	-0.0025
10%TS	0.04421	0.0445	0.0439	0.0443	0.0443	0.0443
20%TS	0.0835	0.0832	0.0827	0.0840	0.0840	0.0840

- Vehicle

Table B.43: *The RLA of the fusion methods for the MI based MCMLPS when applied to the vehicle data set.*

MI based MCMLPS applied to vehicle data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	-0.0163	-0.0277	0.0109	-0.0128	-0.0111	-0.0111
20%TR	-0.0015	-0.0195	-0.0046	-0.0063	-0.0047	-0.0047
10%TS	0.0504	0.0533	0.0478	0.0527	0.0528	0.0528
20%TS	0.1101	0.1168	0.1111	0.1166	0.1181	0.1181

- Waveform

Table B.44: *The RLA of the fusion methods for the MI based MCMLPS when applied to the waveform data set.*

MI based MCMLPS applied to waveform data set						
Noise ratio	Single LR	Best Model	MV	WMV	WMV+accuracy in 1st layers	WMV+accuracy in all layers
10%TR	0.0127	-0.0078	0.0048	-0.0037	-0.0041	-0.0044
20%TR	0.0037	-0.0101	-0.0123	-0.0093	-0.0071	-0.0056
10%TS	0.0515	0.0525	0.0500	0.0522	0.0530	0.0533
20%TS	0.1042	0.1020	0.1026	0.1027	0.1024	0.1029

References

Al-Ani, A. and Deriche, M. (2002). A new technique for combining multiple classifiers using the dempster-shafer theory of evidence. *Journal of Artificial Intelligence Research*, 17:333–361.

Al-Jubouri, B. and Gabrys, B. (2014). Multicriteria approaches for predictive model generation: a comparative experimental study. In *Computational Intelligence in Multi-Criteria Decision-Making (MCDM), 2014 IEEE Symposium on*, pages 64–71. IEEE.

Al-Jubouri, B. and Gabrys, B. (2016). Local learning for multi-layer, multi-component predictive system. *Procedia Computer Science*, 96:723–732.

Al-Jubouri, B. and Gabrys, B. (2017). Diversity and locality in multi-component, multi-layer predictive systems: A mutual information based approach. In *International Conference on Advanced Data Mining and Applications*, pages 313–325. Springer.

Alpaydin, E. (2014). *Introduction to machine learning*. MIT press.

Andersson, J. (2000). A survey of multiobjective optimization in engineering design. *Department of Mechanical Engineering, Linktjping University. Sweden*.

Anguera, X., Shinozaki, T., Wooters, C., and Hernando, J. (2007). Model complexity selection and cross-validation em training for robust speaker diarization. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–273. IEEE.

Anzai, Y. (2012). *Pattern recognition and machine learning*. Elsevier.

Barron, A. R. (1991). Complexity regularization with application to artificial neural networks. In *Nonparametric functional estimation and related topics*, pages 561–576. Springer.

Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, 5(4):537–550.

Bell, A. J. and Sejnowski, T. J. (1997). The independent components of natural scenes are edge filters. *Vision research*, 37(23):3327–3338.

Bertsimas, D. and Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical programming*, 98(1-3):49–71.

Bertsimas, D. and Sim, M. (2004). The price of robustness. *Operations research*, 52(1):35–53.

Bertsimas, D. and Thiele, A. (2006). Robust and data-driven optimization: modern decision making under uncertainty. In *Models, Methods, and Applications for Innovative Decision Making*, pages 95–122. INFORMS.

Bi, Y. (2012). The impact of diversity on the accuracy of evidential classifier ensembles. *International Journal of Approximate Reasoning*, 53(4):584–607.

Birge, J. R. and Louveaux, F. (2011). *Introduction to stochastic programming*. Springer.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Bloch, I. (1996). Information combination operators for data fusion: A comparative review with classification. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 26(1):52–67.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1987). Occam's razor. *Information processing letters*, 24(6):377–380.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526.

Bowman, A. W. and Azzalini, A. (1997). *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations: The Kernel Approach with S-Plus Illustrations*. Oxford University Press.

Braga, A. P., Takahashi, R., Costa, M. A., and de Albuquerque Teixeira, R. (2006). Multi-objective algorithms for neural networks learning. In *Multi-objective machine learning*, pages 151–171. Springer.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Brown, G. and Kuncheva, L. I. (2010). good and bad diversity in majority vote ensembles. In *International Workshop on Multiple Classifier Systems*, pages 124–133. Springer.

Brown, G., Pocock, A., Zhao, M.-J., and Luján, M. (2012). Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *Journal of Machine Learning Research*, 13(Jan):27–66.

Brown, G., Wyatt, J., Harris, R., and Yao, X. (2005). Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20.

Bruha, I. and Famili, A. (2000). Postprocessing in machine learning and data mining. *ACM SIGKDD Explorations Newsletter*, 2(2):110–114.

Budka, M. (2010). *Physically inspired methods and development of data-driven predictive systems.* PhD thesis, Bournemouth University.

Budka, M. and Gabrys, B. (2013). Density-preserving sampling: robust and efficient alternative to cross-validation for error estimation. *Neural Networks and Learning Systems, IEEE Transactions on*, 24(1):22–34.

Budka, M., Gabrys, B., and Ravagnan, E. (2010). Robust predictive modelling of water pollution using biomarker data. *Water research*, 44(10):3294–3308.

Caramanis, C., Mannor, S., and Xu, H. (2011). 14 robust optimization in machine learning. *Optimization for machine learning*, page 369.

Cervier, D. (1993). Ai: The tumultuous search for artificial intelligence.

Coates, A. and Ng, A. Y. (2011). Selecting receptive fields in deep networks. In *Advances in Neural Information Processing Systems*, pages 2528–2536.

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

Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

Cunningham, P. and Carney, J. (2000). Diversity versus quality in classification ensembles based on feature selection. In *European Conference on Machine Learning*, pages 109–116. Springer.

Dan Foresee, F. and Hagan, M. T. (1997). Gauss-newton approximation to bayesian

learning. In *Neural Networks, 1997., International Conference on*, volume 3, pages 1930–1935. IEEE.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., New York, NY, USA.

Deb, K. (2003). Multi-objective evolutionary algorithms: Introducing bias among pareto-optimal solutions. In *Advances in evolutionary computing*, pages 263–292. Springer.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Deng, L. and Yu, D. (2014). Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387.

Deriche, M. and Al-Ani, A. (2001). A new algorithm for eeg feature selection using mutual information. In *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, volume 2, pages 1057–1060. IEEE.

Domingos, P. (2000). A unified bias-variance decomposition. In *Proceedings of 17th International Conference on Machine Learning. Stanford CA Morgan Kaufmann*, pages 231–238.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

Dreyer, J. (1890). *Tycho Brahe: A Picture of Scientific Life and Work in the Sixteenth Century 1890*. Kessinger Publishing.

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*. John Wiley & Sons.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Citeseer.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

Fortmann-Roe, S. (2012). Understanding the bias-variance tradeoff.

Freitas, A. A. (2004a). A critical review of multi-objective optimization in data mining: a position paper. *SIGKDD Explor. Newsl.*, 6(2):77–86.

Freitas, A. A. (2004b). A critical review of multi-objective optimization in data mining: a position paper. *ACM SIGKDD Explorations Newsletter*, 6(2):77–86.

Frénay, B. and Verleysen, M. (2014). Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58.

Gunn, S. R. (2012). Introduction to machine learning. In *Lecture note*, Lecture Notes in Adavance machine learning. School of Electronics and Computer Science,University of Southampton.

Guo, B. and Nixon, M. S. (2009). Gait feature subset selection by mutual information. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39(1):36–46.

Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. A. (2008). *Feature extraction: foundations and applications*, volume 207. Springer.

Hagan, M. T. and Menhaj, M. B. (1994). Training feedforward networks with the marquardt algorithm. *Neural Networks, IEEE Transactions on*, 5(6):989–993.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hastie, T., Tibshirani, R., and Friedman, J. (2002). The elements of statistical learning: Data mining, inference, and prediction. *Biometrics*.

Hatami, N. and Ebrahimpour, R. (2007). Combining multiple classifiers: diversify with boosting and combining by stacking. *International Journal of Computer Science and Network Security*, 7(1):127–131.

Hatanaka, T., Kondo, N., and Uosaki, K. (2003). Multi-objective structure selection for radial basis function networks based on genetic algorithm. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 1095–1100. IEEE.

Hernández-Lobato, J. M. (2010). Balancing flexibility and robustness in machine learning: semi-parametric methods and sparse linear models.

Hickey, R. J. (1996). Noise modelling and evaluating learning from examples. *Artificial Intelligence*, 82(1):157–179.

Ho, T. K. (2001). Data complexity analysis for classifer combination. In *Proceedings of the 2nd International Workshop on Multiple Classi er Systems*, page 53. Citeseer.

Ho, T. K., Hull, J. J., and Srihari, S. N. (1994). Decision combination in multiple classifier systems. *IEEE transactions on pattern analysis and machine intelligence*, 16(1):66–75.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688.

Islam, M. M., Yao, X., and Murase, K. (2003). A constructive algorithm for training cooperative neural network ensembles. *Neural Networks, IEEE Transactions on*, 14(4):820–834.

Jacobs, R. A. (1995). Methods for combining experts' probability assessments. *Neural computation*, 7(5):867–888.

Jankowski, N. and Grabczewski, K. (2011). Universal meta-learning architecture and algorithms. In *Meta-Learning in Computational Intelligence*, pages 1–76. Springer.

Jee, K.-W., McShan, D. L., and Fraass, B. A. (2007). Lexicographic ordering: intuitive multicriteria optimization for imrt. *Physics in Medicine and Biology*, 52(7):1845.

Jen, E. (2003). Stable or robust? what's the difference? *Complexity*, 8(3):12–18.

Jin, Y. (2000). Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. *Fuzzy Systems, IEEE Transactions on*, 8(2):212–221.

Jin, Y. (2006). *Multi-objective machine learning*, volume 16. Springer Science & Business Media.

Jin, Y. and Branke, J. (2005). Evolutionary optimization in uncertain environments-a survey. *Evolutionary Computation, IEEE Transactions on*, 9(3):303–317.

Jin, Y. and Sendhoff, B. (2008). Pareto-based multiobjective machine learning: An

overview and case studies. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(3):397–415.

Juszczak, P. and Duin, R. (2004). Combining one-class classifiers to classify missing data. *Multiple Classifier Systems*, pages 92–101.

Kadlec, P. and Gabrys, B. (2009). Architecture for development of adaptive on-line prediction models. *Memetic Computing*, 1(4):241–269.

Kadlec, P. and Gabrys, B. (2011). Local learning-based adaptive soft sensor for catalyst activation prediction. *AIChE Journal*, 57(5):1288–1301.

Kadlec, P., Grbic, R., and Gabrys, B. (2011). Review of adaptation mechanisms for data-driven soft sensors. *Computers & chemical engineering*, 35(1):1–24.

Kali, P. and Wallace, S. W. (1994). *Stochastic programming*. Springer.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45.

Kearns, M. J. and Schapire, R. E. (1994). Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48(3):464–497.

Kindermann, R. P. and Snell, J. L. (1980). On the relation between markov random fields and social networks. *Journal of Mathematical Sociology*, 7(1):1–13.

King, R. D., Feng, C., and Sutherland, A. (1995). Statlog: comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence an International Journal*, 9(3):289–333.

Ko, A. H., Sabourin, R., and Britto Jr, A. S. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, 41(5):1718–1731.

Kotsiantis, S. (2011). Combining bagging, boosting, rotation forest and random subspace methods. *Artificial Intelligence Review*, 35(3):223–240.

Kotsiantis, S. and Pintelas, P. (2004). Combining bagging and boosting. *International Journal of Computational Intelligence*, 1(4):324–333.

Krause, S. and Polikar, R. (2003). An ensemble of classifiers approach for the missing feature problem. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 1, pages 553–558. IEEE.

Krogh, A. and Vedelsby, J. (1995). Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7:231–238.

Kubat, M., Holte, R. C., and Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, 30(2-3):195–215.

Kuncheva, L. (2000). *Fuzzy classifier design*, volume 49. Springer Science & Business Media.

Kuncheva, L. I. (2004). *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons.

Kuncheva, L. I., Skurichina, M., and Duin, R. P. (2002). An experimental study on diversity for bagging and boosting with linear classifiers. *Information fusion*, 3(4):245–258.

Kuncheva, L. I. and Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207.

Lam, L. (2000). Classifier combinations: implementations and theoretical issues. In *International Workshop on Multiple Classifier Systems*, pages 77–86. Springer.

Langdon, W. B., Barrett, S., and Buxton, B. F. (2002). Combining decision trees and neural networks for drug discovery. In *Genetic Programming*, pages 60–70. Springer.

Lichman, M. (2013). UCI machine learning repository.

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

Marler, R. T. and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 11(2):431–441.

Mazurov, V. D., Krivonogov, A. I., and Kazantsev, V. S. (1987). Solving of optimization and identification problems by the committee methods. *Pattern Recognition*, 20(4):371–378.

Meir, R. (1995). Bias, variance and the combination of estimators; the case of linear least squares. In *In Advances in Neural Information Processing Systems 7*. Citeseer.

Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (2013). *Machine learning: An artificial intelligence approach*. Springer Science & Business Media.

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

Močkus, J. (1975). On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer.

Nauck, D. D. (2003). Measuring interpretability in rule-based classification systems. In *Fuzzy Systems, 2003. FUZZ'03. The 12th IEEE International Conference on*, volume 1, pages 196–201. IEEE.

Nauck, D. D. (2005). Neuro-fuzzy systems for explaining data sets. In *Do Smart Adaptive Systems Exist?*, pages 305–319. Springer.

Opitz, D. W. and Shavlik, J. W. (1996a). Actively searching for an effective neural network ensemble. *Connection Science*, 8(3-4):337–354.

Opitz, D. W. and Shavlik, J. W. (1996b). Generating accurate and diverse members of a neural-network ensemble. *Advances in neural information processing systems*, pages 535–541.

Panov, P. and Dzeroski, S. (2007). Combining bagging and random subspaces to create better ensembles. In *IDA*, volume 7, pages 118–129. Springer.

Parikh, D. and Polikar, R. (2007). An ensemble-based incremental learning approach to data fusion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):437–450.

Parsopoulos, K. E. and Vrahatis, M. N. (2002). Recent approaches to global optimization problems through particle swarm optimization. *Natural computing*, 1(2-3):235–306.

Parvin, H., Minaei-Bidgoli, B., and Shahpar, H. (2011). Classifier selection by clustering. *Pattern Recognition*, pages 60–66.

Polikar, R. (2006). Ensemble based systems in decision making. *Circuits and systems magazine, IEEE*, 6(3):21–45.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier.

Reed, D. (2005). *A balanced introduction to computer science*. Pearson Prentice Hall.

Rentmeesters, M. J., Tsai, W. K., and Lin, K.-J. (1996). A theory of lexicographic multi-

criteria optimization. In *Engineering of Complex Computer Systems, 1996. Proceedings., Second IEEE International Conference on*, pages 76–79. IEEE.

Riedel, S. and Gabrys, B. (2009). Pooling for combination of multilevel forecasts. *Knowledge and Data Engineering, IEEE Transactions on*, 21(12):1753–176.

Ripley, B. D. (2007). *Pattern recognition and neural networks*. Cambridge university press.

Rodriguez, J. J., Kuncheva, L. I., and Alonso, C. J. (2006). Rotation forest: A new classifier ensemble method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1619–1630.

Roli, F., Giacinto, G., and Vernazza, G. (2001). Methods for designing multiple classifier systems. In *International Workshop on Multiple Classifier Systems*, pages 78–87. Springer.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall series in artificial intelligence. Prentice Hall.

Ruta, D. (2003). *Classifier diversity in combined pattern recognition systems*. University of Paisley.

Ruta, D. and Gabrys, B. (2000). An overview of classifier fusion methods. *Computing and Information systems*, 7(1):1–10.

Ruta, D. and Gabrys, B. (2002). A theoretical analysis of the limits of majority voting errors for multiple classifier systems. *Pattern Analysis & Applications*, 5(4):333–350.

Ruta, D. and Gabrys, B. (2005). Classifier selection for majority voting. *Information fusion*, 6(1):63–81.

Ruta, D., Gabrys, B., and Lemke, C. (2011). A generic multilevel architecture for time series prediction. *Knowledge and Data Engineering, IEEE Transactions on*, 23(3):350–359.

Sáez, J. A., Galar, M., Luengo, J., and Herrera, F. (2013). Tackling the problem of classi-

fication with noisy data using multiple classifier systems: Analysis of the performance and robustness. *Information Sciences*, 247:1–20.

Salvador, M. M., Budka, M., and Gabrys, B. (2016). Effects of change propagation resulting from adaptive preprocessing in multicomponent predictive systems. In *KES-2016. 20th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems*.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229.

Saul, L. K. and Roweis, S. T. (2000). An introduction to locally linear embedding. *unpublished. Available at: http://www. cs. toronto. edu/˜ roweis/lle/publications. html*.

Shapley, L. and Grofman, B. (1984). Optimizing group judgmental accuracy in the presence of interdependencies. *Public Choice*, 43(3):329–343.

Sharpe, P. K. and Solly, R. (1995). Dealing with missing values in neural network-based diagnostic systems. *Neural Computing & Applications*, 3(2):73–77.

Small, R. (1804). a account of the astronomical discoveries of kepler. *London, Printed for J. Mawman, 1804*.

Suttorp, T. and Igel, C. (2006). Multi-objective optimization of support vector machines. In *Multi-objective machine learning*, pages 199–220. Springer.

Swerdlow, N. M. (1996). Astronomy in the renaissance. In *Astronomy before the Telescope*, pages 187–230.

Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

Teng, C. M. (1999). Correcting noisy data. In *Machine Learning*. Citeseer.

Theodoridis, S. and Koutroumbas, K. (2006). *Pattern Recognition, Third Edition*. Academic Press, Inc., Orlando, FL, USA.

Theodoridis, S., Pikrakis, A., Koutroumbas, K., and Cavouras, D. (2010). *Introduction to pattern recognition: a matlab approach*. Academic Press.

Toivonen, H., Klemettinen, M., Ronkainen, P., Hätönen, K., and Mannila, H. (1995). Pruning and grouping discovered association rules.

Tumer, K. and Ghosh, J. (1996). Error correlation and error reduction in ensemble classifiers. *Connection science*, 8(3-4):385–404.

Tumer, K. and Oza, N. C. (2003). Input decimated ensembles. *Pattern Analysis & Applications*, 6(1):65–77.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Vapnik, V. N. and Kotz, S. (1982). *Estimation of dependences based on empirical data*, volume 41. Springer-Verlag New York.

Wang, W., Jones, P., and Partridge, D. (2000). Diversity between neural networks and decision trees for building multiple classifier systems. In *Multiple Classifier Systems*, pages 240–249. Springer.

Wasserman, L. (2006). Nonparametric inference using orthogonal functions. *All of Nonparametric Statistics*, pages 183–195.

Wasserman, L. (2013). *All of statistics: a concise course in statistical inference*. Springer Science & Business Media.

Whalen, S. and Pandey, G. (2013). A comparative analysis of ensemble classifiers: case studies in genomics. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 807–816. IEEE.

Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259.

Woods, K., Kegelmeyer, W. P., and Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE transactions on pattern analysis and machine intelligence*, 19(4):405–410.

Woźniak, M., Graña, M., and Corchado, E. (2014). A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17.

Xu, H., Caramanis, C., and Mannor, S. (2009). Robust regression and lasso. In *Advances in Neural Information Processing Systems*, pages 1801–1808.

Xue, F., Subbu, R., and Bonissone, P. (2006). Locally weighted fusion of multiple pre-

dictive models. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 2137–2143. IEEE.

Yu, L., Lai, K. K., Wang, S., and Huang, W. (2006). A bias-variance-complexity trade-off framework for complex system modeling. In *Computational Science and Its Applications-ICCSA 2006*, pages 518–527. Springer.

Zhang, C. and Ma, Y. (2012). *Ensemble machine learning: methods and applications*. Springer.

Zhang, C.-X. and Zhang, J.-S. (2009). A novel method for constructing ensemble classifiers. *Statistics and Computing*, 19(3):317–327.

Zhang, Z. and Yang, P. (2008). An ensemble of classifiers with genetic algorithmbased feature selection. *The IEEE intelligent informatics bulletin*, 9(1):18–24.

Zhao, H. (2007). A multi-objective genetic programming approach to developing pareto optimal decision trees. *Decision Support Systems*, 43(3):809–826.

Zhou, G. and Si, J. (1998). Advanced neural-network training algorithm with reduced complexity based on jacobian deficiency. *Neural Networks, IEEE Transactions on*, 9(3):448–453.

Zhu, X. and Wu, X. (2004). Class noise vs. attribute noise: A quantitative study. *Artificial Intelligence Review*, 22(3):177–210.

Zitzler, E., Laumanns, M., and Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. Technical report.

Žliobaitė, I., Budka, M., and Stahl, F. (2015). Towards cost-sensitive adaptation: When is it worth updating your predictive model? *Neurocomputing*, 150:240–249.

Zliobaite, I. and Gabrys, B. (2014). Adaptive preprocessing for streaming data. *IEEE transactions on knowledge and data Engineering*, 26(2):309–321.